בס"ד

מיני פרויקט בבסיסי נתונים

ניהול ספריה באוניברסיטה

מגיש: עדן ישי כהן

תוכן עניינים:

שלב שלישי

הקדמה:

החלק הזה כולל שני חלקים עיקריים שנועדו לייעל את ניהול המשימות והשאלת הספרים בספרייה. החלקים מחולקים כך שיספקו פתרון לניהול יומיומי של משימות הספרייה והשירותים הניתנים לסטודנטים:

1. **ניהול שיבוצי משימות בספרייה:**

חלק זה מתמקד במערכת שיבוץ משימות שמאפשרת לספרנים ליצור ולנהל משימות חדשות באופן יעיל. כל משימה שנוצרת יכולה להיות מטופלת באחת משלוש אפשרויות שיבוץ שונות, כדי להתאים לדרישות התפעוליות של הספרייה:

- **שיבוץ ידני :**הספרן שיוצר את המשימה בוחר בעצמו מי יהיה הספרן שישובץ למשימה. אפשרות זו מתאימה למקרים בהם יש צורך בידע מקצועי מסוים או כאשר נדרש ספרן בעל התמחות ספציפית
- **שיבוץ אוטומטי :**הספרן שיוצר את המשימה מגדיר שהשיבוץ יתבצע אוטומטית על סמך ספרן פנוי. השיבוץ מתבצע על פי הספרן שביצע את מספר המשימות הקטן ביותר עד כה, ובהתאם לתאריך העסקתו במקרה של תיקו. אפשרות זו מתאימה לניהול עומסים והבטחת חלוקה הוגנת של המשימות.
- **משימה בהמתנה :**הספרן מגדיר שהמשימה תישאר במצב "ממתין" ללא שיבוץ מיידי. אפשרות זו מתאימה כאשר לא ניתן לשבץ משימה לספרן באופן מיידי, או כאשר המשימה דורשת שיבוץ עתידי

- **בדיקות והדגמות:**

ביצעתי הרצה של התוכנית הראשית לכל אחת משלוש האפשרויות עם פרמטרים מותאמים, כדי להבטיח שהמערכת פועלת בצורה תקינה בכל אחד מהמקרים. כמו כן, בדקתי את השפעת הבחירות על נתוני הטבלאות (כגון LibraryRequests ו-RequestAssignments), ווידאתי שכל המשימות משובצות כראוי.

2. **ניהול השאלות והחזרות ספרים:**

חלק זה מתמקד בניהול תהליך ההשאלה והחזרת הספרים לסטודנטים, כולל חישובי קנסות ועדכון סטטוס הספרים במערכת.

שני המקרים המרכזיים:

- **תוכנית ראשית לניהול השאלת ספרים:** מערכת ההשאלה נועדה להבטיח שהסטודנטים יוכלו לשאול ספרים בצורה מאורגנת ותוך שמירה על מגבלות מסוימות. התוכנית הראשית מטפלת בכל ההיבטים של השאלת ספרים: בדיקה אם הסטודנט חרג ממגבלת החובות המותרת (200 שקלים), בדיקה שהסטודנט לא עבר את מגבלת השאלת הספרים (10 ספרים), ווידוא שמשכי ההשאלה לכל ספר עומדים בגבולות ההרשאה (עד 30 יום).
- **תוכנית ראשית לניהול החזרת ספרים :**מערכת החזרת הספרים מטפלת בעדכון סטטוס הספרים ובחישוב הקנסות בהתאם למצב הספר: קנס על איחור בהחזרה בהתאם למספר ימי האיחור, קנס קבוע עבור ספרים שאבדו, עדכון נתוני הטבלאות BookStatuses- Lent בהתאם לתהליך ההחזרה.

- **בדיקות והדגמות:**

ביצעתי הרצה של התוכניות הראשיות הן להשאלת ספרים והן להחזרת ספרים, תוך שימוש בפרמטרים רלוונטיים. במהלך ההרצה נבדקו השפעות התהליכים על הטבלאות (Lent ו-BookStatuses) ווידאתי שכל התהליכים מתנהלים כמצופה, כולל טיפול בחריגות כמו סטודנטים עם חובות או ספרים שאבדו.

בס"ד

ניהול שיבוצי משימות בספרייה:

פונקציה 1: get_least_busy_librarian

מטרת הפונקציה:
הפונקציה נועדה להחזיר את מזהה הספרן הפנוי ביותר על בסיס מספר המשימות המשובצות לו. במידה ויש תיקו במספר המשימות, הספרן עם תאריך ההעסקה המוקדם ביותר ייבחר.

הסבר הפונקציה:
הפונקציה בודקת תחילה אם קיימים ספרנים במערכת. לאחר מכן, היא משתמשת ב-CURSOR למציאת הספרן הפנוי ביותר, תוך סידור לפי מספר המשימות ותאריך ההעסקה. במקרה שבו אין נתונים, מתבצע טיפול בחריגות מתאימות.

```oracle_plsql
CREATE OR REPLACE FUNCTION get_least_busy_librarian
RETURN INT IS
    v_librarian_id INT;
BEGIN
    -- Check if there are any librarians in the system
    DECLARE
        v_librarians_count INT;
    BEGIN
        SELECT COUNT(*) INTO v_librarians_count FROM Librarians;
        IF v_librarians_count = 0 THEN
            DBMS_OUTPUT.PUT_LINE('Error: No librarians found in the system.');
            RETURN NULL;
        END IF;
    END;

    -- Find the least busy librarian based on the number of assigned tasks and hire date
    DECLARE
        CURSOR librarian_cursor IS
            SELECT l.Librarian_id
            FROM Librarians l
            LEFT JOIN RequestAssignments ra ON l.Librarian_id = ra.Assigned_librarian
            GROUP BY l.Librarian_id, l.Hire_date
            ORDER BY COUNT(ra.Request_id) ASC, l.Hire_date ASC;

    BEGIN
        OPEN librarian_cursor;
        FETCH librarian_cursor INTO v_librarian_id;
        CLOSE librarian_cursor;

        RETURN v_librarian_id;
    EXCEPTION
        WHEN NO_DATA_FOUND THEN
            DBMS_OUTPUT.PUT_LINE('Error: No available librarian found.');
            RETURN NULL;
    END;
END;
```

פונקציה 2: is_librarian_assigned

מטרת הפונקציה:
הפונקציה נועדה לבדוק אם ספרן מסוים מוקצה כרגע למשימות בעלות סטטוס "in progress".

הסבר הפונקציה:
הפונקציה מקבלת מזהה ספרן כפרמטר ובודקת אם הוא משובץ למשימות פעילות. היא מטפלת במקרים שבהם מזהה הספרן אינו חוקי ומחזירה TRUE אם הספרן מוקצה למשימות ו-FALSE אם לא.

```oracle_plsql
CREATE OR REPLACE FUNCTION is_librarian_assigned(v_librarian_id INT)
RETURN BOOLEAN IS
    v_count INT;
BEGIN
    -- Check if the librarian ID is valid
    IF v_librarian_id IS NULL OR v_librarian_id <= 0 THEN
        DBMS_OUTPUT.PUT_LINE('Error: Invalid librarian ID.');
        RETURN FALSE;
    END IF;

    -- Check if the librarian is assigned to any tasks - return TRUE if assigned, FALSE otherwise
    SELECT COUNT(*)
    INTO v_count
    FROM RequestAssignments ra
    JOIN LibraryRequests lr ON ra.request_id = lr.request_id
    WHERE ra.assigned_librarian = v_librarian_id
    AND lr.request_status = 'in progress';

    RETURN v_count > 0;
END;
```

פרוצדורה 1: assign_librarian_to_task

מטרת הפרוצדורה:
הפרוצדורה משבצת ספרן למשימה מסוימת, בהתאם למזהה משימה שהוזן כפרמטר. אם לא סופק מזהה ספרן, מתבצע שיבוץ אוטומטי לספרן הפנוי ביותר.

הסבר הפרוצדורה:
הפרוצדורה בודקת את תקינות מזהי המשימה והספרן, ואם לא נבחר ספרן באופן ידני, היא מזמנת את הפונקציה get_least_busy_librarian לבחירת ספרן אוטומטית. לאחר מכן, מתבצע עדכון סטטוס המשימה ל-"in progress" והוספת הרשומה לטבלת המשימות המשובצות.

```oracle_plsql
CREATE OR REPLACE PROCEDURE assign_librarian_to_task(
    v_task_id INT,
    v_librarian_id INT -- For automatic assignment, leave NULL
) IS
    v_chosen_librarian INT;
    v_librarian_exists INT;
BEGIN
    -- Check if the task ID is valid
    IF v_task_id IS NULL OR v_task_id <= 0 THEN
        DBMS_OUTPUT.PUT_LINE('Error: Invalid task ID.');
        RETURN;
    END IF;

    -- Check existence of the librarian in the system
    IF v_librarian_id IS NOT NULL THEN
        SELECT COUNT(*) INTO v_librarian_exists
        FROM Librarians
        WHERE Librarian_id = v_librarian_id;

        IF v_librarian_exists = 0 THEN
            DBMS_OUTPUT.PUT_LINE('Error: Assigned librarian does not exist in the system.');
            RETURN;
        END IF;

        v_chosen_librarian := v_librarian_id;
    ELSE
        -- Automatic assignment of the least busy librarian
        v_chosen_librarian := get_least_busy_librarian;
        IF v_chosen_librarian IS NULL THEN
            DBMS_OUTPUT.PUT_LINE('Error: Unable to assign a librarian.');
            RETURN;
        END IF;
    END IF;

    -- Update the task to 'in progress' status
    UPDATE LibraryRequests
    SET request_status = 'in progress'
    WHERE request_id = v_task_id;

    -- Insert the assignment into the RequestAssignments table
    INSERT INTO RequestAssignments (request_id, assigned_librarian)
    VALUES (v_task_id, v_chosen_librarian);

    DBMS_OUTPUT.PUT_LINE('Librarian ID ' || v_chosen_librarian || ' assigned to task ' || v_task_id || '.');
END;
```

פרוצדורה 2: create_new_task

מטרת הפרוצדורה:
הפרוצדורה יוצרת משימה חדשה עבור הספרייה עם פרטים כגון סוג המשימה, תיאור המשימה, תאריך יצירה ומזהה הספרן שיצר את המשימה.

הסבר הפרוצדורה:
הפרוצדורה בודקת את תקינות הפרמטרים שהוזנו, כגון סוג ותיאור המשימה, מזהה הספרן ותאריך המשימה. היא מחשבת מזהה משימה חדש באופן דינמי ומוסיפה רשומה חדשה לטבלת המשימות.

```oracle_plsql
CREATE OR REPLACE PROCEDURE create_new_task(
    v_request_type VARCHAR2,
    v_request_description CLOB,
    v_request_date DATE,
    v_request_librarian INT -- ID of the librarian creating the task
) IS
    v_new_request_id INT;
    v_creator_exists INT;
BEGIN
    -- Check if the input parameters are valid
    IF v_request_type IS NULL OR LENGTH(v_request_type) = 0 THEN
        DBMS_OUTPUT.PUT_LINE('Error: Request type cannot be null or empty.');
        RETURN;
    END IF;

    IF v_request_description IS NULL OR LENGTH(v_request_description) = 0 THEN
        DBMS_OUTPUT.PUT_LINE('Error: Request description cannot be null or empty.');
        RETURN;
    END IF;

    IF v_request_librarian IS NULL OR v_request_librarian <= 0 THEN
        DBMS_OUTPUT.PUT_LINE('Error: Invalid librarian ID.');
        RETURN;
    END IF;

    -- Check if the creating librarian exists in the system
    SELECT COUNT(*) INTO v_creator_exists
    FROM Librarians
    WHERE Librarian_id = v_request_librarian;

    IF v_creator_exists = 0 THEN
        DBMS_OUTPUT.PUT_LINE('Error: Creating librarian does not exist in the system.');
        RETURN;
    END IF;

    -- Calculate the new request ID by finding the maximum and adding 1
    SELECT NVL(MAX(request_id), 0) + 1 INTO v_new_request_id FROM LibraryRequests;

    -- Insert a new task into the LibraryRequests table with the calculated request ID
    INSERT INTO LibraryRequests (request_id, request_type, request_description, request_date, request_status, request_librarian)
    VALUES (v_new_request_id, v_request_type, v_request_description, v_request_date, 'waiting', v_request_librarian);

    DBMS_OUTPUT.PUT_LINE('New task created with ID: ' || v_new_request_id);
END;
```

תוכנית ראשית:

מטרת התוכנית

מטרת התוכנית היא לנהל את תהליך יצירת משימות חדשות במערכת הספרייה ושיבוצן לספרנים. התוכנית מאפשרת גמישות בכך שהיא תומכת בשלושה תרחישים שונים: שיבוץ ידני של ספרן מסוים למשימה, שיבוץ אוטומטי לספרן הפנוי ביותר, והשארת המשימה ללא שיבוץ בסטטוס "ממתין". בצורה זו, ניתן לנהל את המשימות בצורה יעילה תוך שמירה על איזון בין העומס על הספרנים והצרכים של הספרייה.

הסבר התוכנית

בתחילת התוכנית מתבצע קריאה לפונקציה create_new_task אשר יוצרת משימה חדשה במערכת. הפרטים של המשימה כמו סוג המשימה, תיאור, תאריך יצירה ומזהה הספרן שיוצר את המשימה מסופקים על ידי המשתמש. לאחר יצירת המשימה, מזהה המשימה שנוצר נשמר באמצעות שאילתת SQL אשר מחזירה את המזהה הגבוה ביותר בטבלת המשימות.

התוכנית ממשיכה לפי בחירת המשתמש באפשרות השיבוץ. אם נבחר שיבוץ ידני, המשימה משובצת לספרן שמספר המזהה שלו סופק מראש. אם נבחר שיבוץ אוטומטי, המשימה משובצת לספרן הפנוי ביותר במערכת על ידי קריאה לפונקציה מתאימה. אם נבחר להשאיר את המשימה במצב "ממתין", לא מתבצע שיבוץ והמשימה נשארת ללא ספרן משובץ. אם הבחירה של המשתמש אינה תקפה, התוכנית מדפיסה הודעה מתאימה ומציינת שהשיבוץ לא בוצע.

במהלך הפעולה התוכנית מדפיסה הודעות שמאשרות את יצירת המשימה, בחירת שיטת השיבוץ וביצוע השיבוץ עצמו. במידה וקיימות בעיות, התוכנית מציינת זאת בהודעות מודפסות.

```oracle_plsql
DECLARE
    v_request_type VARCHAR2(50) := 'books'; -- Task type
    v_request_description CLOB := 'Need more books for the children section'; -- Task description
    v_request_date DATE := SYSDATE; -- Task creation date
    v_request_librarian INT := 291563328; -- Librarian creating the task
    v_librarian_choice INT := 1;  -- 1 = manual assignment, 2 = automatic assignment, 3 = no assignment (waiting)
    v_librarian_id INT := 200885937;   -- Specific librarian ID for manual assignment (only relevant for choice 1)
    v_task_id INT; -- Will store the new task ID

BEGIN
    -- Create a new task
    create_new_task(v_request_type, v_request_description, v_request_date, v_request_librarian);

    -- Retrieve the ID of the newly created task
    SELECT MAX(request_id) INTO v_task_id FROM LibraryRequests;

    -- Handle assignment choice based on v_librarian_choice
    IF v_librarian_choice = 1 THEN
        -- Manual assignment of a librarian by ID
        assign_librarian_to_task(v_task_id, v_librarian_id);
        DBMS_OUTPUT.PUT_LINE('Task ' || v_task_id || ' manually assigned to librarian ID ' || v_librarian_id || '.');

    ELSIF v_librarian_choice = 2 THEN
        -- Automatic assignment to the least busy librarian
        assign_librarian_to_task(v_task_id, NULL);
        DBMS_OUTPUT.PUT_LINE('Task ' || v_task_id || ' automatically assigned to the least busy librarian.');

    ELSIF v_librarian_choice = 3 THEN
        -- Leave the task in "waiting" status with no librarian assigned
        DBMS_OUTPUT.PUT_LINE('Task ' || v_task_id || ' created with no librarian assignment (waiting status).');

    ELSE
        DBMS_OUTPUT.PUT_LINE('Invalid choice. Task was not assigned.');
    END IF;

END;
```

כמות כל המשימות שבוצעו, מבוצעים, נסגרו או טרם שובץ אליהן ספרן.

	COUNT(*)
1	2000

```oracle_plsql
-- Query to get the number of tasks:
select count(*) from libraryrequests;
```

כמות כל המשימות פרט לאלה שטרם שובץ אליהן ספרן.

	COUNT(*)
1	1481

```oracle_plsql
-- Query to get the number of assigments request:
select count(*) from requestassignments;
```

הרצת Main ראשונה: הפרמטר v_librarian_choice שווה ל-1
(כלומר הספרן שיוצר את המשימה הוא גם יציין את הספרן שישובץ אל המשימה)

```
New task created with ID: 2001
Librarian ID 200885937 assigned to task 2001.
Task 2001 manually assigned to librarian ID 200885937.


PL/SQL procedure successfully completed.
```

הודפס שנוצרה משימה חדשה עם מזהה שמספרו 2001 שאותה יצר הספרן עם המזהה 291563328. בנוסף הודפס שהמשימה שובצה בהצלחה לספרן 200885937 שצוין בפרמטר.

בדיקה של הנתונים החדשים:

לטבלה של LibraryRequest התווספה משימה חדשה בדיוק כמו שצוין בפרמטרים. וגם הסטטוס הוא "in progress" שזה אומר שכבר שובץ אליה ספרן אבל עדיין לא הסתיימה או נסגרה.

```
-- Query to get the number of tasks:
select count(*) from libraryrequests;
```

	COUNT(*)
1	2001

```
-- -- Query to get all the new library tasks
select * from libraryrequests
where request_id=2001
order by request_id;
```

pt Output | Query Result

SQL | All Rows Fetched: 1 in 0.003 seconds

	REQUEST_ID	REQUEST_TYPE	REQUEST_DESCRIPTION	REQUEST_DATE	REQUEST_LIBRARIAN	REQUEST_STATUS
1	2001	books	Need more books for the children section	29-11-2024 02:01:52	291563328	in progress

בטבלה של requestassignments אפשר לראות ששובץ אליה הספרן מהפרמטר ושאין לה עדיין תאריך סיום. (משימות שטרם שובץ אליהם ספרן לא יופיעו בטבלה הזאת אלא יהיה רק בטבלה הראשונה תחת הסטטוס waiting".

```
-- Query to get the number of assigments request:
select count(*) from requestassignments;
```

	COUNT(*)
1	1482

```
-- -- Query to get all the new library tasks
select * from requestassignments
where request_id>2000
order by request_id;
```

ript Output | Query Result

SQL | All Rows Fetched: 1 in 0.002 seconds

	REQUEST_ID	ASSIGNED_LIBRARIAN	COMPLETION_DATE
1	2001	200885937	(null)

הרצת Main שנייה - הפרמטר v_librarian_choice שווה ל-2
(כלומר הספרן שישובץ למשימה ישובץ באופן אוטומטי)

```
DECLARE
    v_request_type VARCHAR2(50) := 'books';  -- Task type
    v_request_description CLOB := 'Need more books for the children section';  -- Task description
    v_request_date DATE := SYSDATE;  -- Task creation date
    v_request_librarian INT := 291563328;  -- Librarian creating the task
    v_librarian_choice INT := 2;  -- 1 = manual assignment, 2 = automatic assignment, 3 = no assignment (waiting)
    v_librarian_id INT := NULL;  -- Specific librarian ID for manual assignment (only relevant for choice 1)
    v_task_id INT;  -- Will store the new task ID
```

אחרי ההרצה קיבלתי את ההדפסה הזאת שנוצרה המשימה 2001 ע"י ספרן 291563328 ושהיא קיבלה שיבוץ.

```
New task created with ID: 2001
Librarian ID 397795675 assigned to task 2001.
Task 2001 automatically assigned to the least busy librarian.


PL/SQL procedure successfully completed.
```

בדיקה של הנתונים החדשים:

אז אפשר לראות בתמונה הבאה שהמשימה נוצרה בהצלחה

```
-- Query to get the number of tasks:
select count(*) from libraryrequests;
```

	COUNT(*)
1	2001

```
-- -- Query to get all the new library tasks
select * from libraryrequests
where request_id=2001
order by request_id;
```

pt Output × | Query Result ×

SQL | All Rows Fetched: 1 in 0.003 seconds

	REQUEST_ID	REQUEST_TYPE	REQUEST_DESCRIPTION	REQUEST_DATE	REQUEST_LIBRARIAN	REQUEST_STATUS
1	2001	books	Need more books for the children section	29-11-2024 02:01:52	291563328	in progress

והמשימה קיבלה שיבוץ ע"י הספרן עם המזהה 397795675 מהתמונה הבאה:

```
-- Query to get the last task:
select * from requestassignments
WHERE request_id = (SELECT MAX(request_id) FROM LibraryRequests);
```

ipt Output × | Query Result ×

SQL | All Rows Fetched: 1 in 0.001 seconds

	REQUEST_ID	ASSIGNED_LIBRARIAN	COMPLETION_DATE
1	2001	397795675	(null)

```
-- Query to get the number of assigments request:
select count(*) from requestassignments;
```

	COUNT(*)
1	1482

הרצת Main שלישית - הפרמטר v_librarian_choice שווה ל-3
(כלומר, המשימה נותרת אבל ללא שיבוץ):

```
DECLARE
    v_request_type VARCHAR2(50) := 'books';  -- Task type
    v_request_description CLOB := 'Need more books for the children section';  -- Task description
    v_request_date DATE := SYSDATE;  -- Task creation date
    v_request_librarian INT := 291563328;  -- Librarian creating the task
    v_librarian_choice INT := 3;  -- 1 = manual assignment, 2 = automatic assignment, 3 = no assignment (waiting)
    v_librarian_id INT := NULL;  -- Specific librarian ID for manual assignment (only relevant for choice 1)
    v_task_id INT;  -- Will store the new task ID
```

בהדפסה הזאת לעומת ההדפסות הקודמות, אין חלק שמציין שהמשימה הזאת קיבלה שיבוץ

```
New task created with ID: 2001
Task 2001 created with no librarian assignment (waiting status).


PL/SQL procedure successfully completed.
```

בדיקה של הנתונים החדשים:

כאן אפשר לראות את המשימה שהיא נוצרה ומכיוון שאין לה שיבוץ אז הסטטוס הוא "waiting".

```
-- Query to get the last task:
select * from libraryrequests where request_id =
(select MAX(request_id) from libraryrequests);
```

t Output × | Query Result ×

SQL | All Rows Fetched: 1 in 0.001 seconds

REQUEST_ID	REQUEST_TYPE	REQUEST_DESCRIPTION	REQUEST_DATE	REQUEST_LIBRARIAN	REQUEST_STATUS
2001	books	Need more books for the children section	29-11-2024 02:34:03	291563328	waiting

	COUNT(*)
1	2001

```
-- Query to get the number of tasks:
select count(*) from libraryrequests;
```

וכאן אפשר לראות שכמות המשימות שקיבלו שיבוץ לא גדלה בכלל.

```
-- Query to get the number of assigments request:
select count(*) from requestassignments;
```

cript Output × | Query Result ×

SQL | All Rows Fetched: 1 in 0.001 seconds

	COUNT(*)
1	1481

ניהול השאלות והחזרות ספרים:

פונקציה 1: calculate_fine

מטרת הפונקציה:
להחזיר את הקנס המתאים עבור ספר שנמצא במצב מסוים ("damaged", "good", או "lost") ולחשב קנס נוסף אם הספר הוחזר באיחור.

הסבר הפונקציה:
הפונקציה מקבלת את מצב הספר (כגון "good" או "damaged") ואת תאריך ההחזרה הצפוי של הספר. אם הספר במצב "lost", הפונקציה מחזירה קנס קבוע של 100. עבור מצבים אחרים, היא מחשבת את מספר ימי האיחור ואת הקנס בהתאם. במצב "damaged", מתווסף קנס בסיסי של 50 עם תוספת לכל יום איחור, ובמצב "good" הקנס מחושב רק על סמך האיחור.

```oracle_plsql
CREATE OR REPLACE FUNCTION calculate_fine(
    v_condition IN VARCHAR2,
    v_return_date IN DATE
) RETURN NUMBER IS
    v_days_late NUMBER := 0;
    v_fine NUMBER := 0;
BEGIN
    IF v_condition NOT IN ('good', 'damaged', 'lost') THEN
        DBMS_OUTPUT.PUT_LINE('Invalid condition specified for the book.');
        RETURN 0;
    END IF;

    IF v_condition = 'lost' THEN
        RETURN 100;
    END IF;

    v_days_late := GREATEST(0, TRUNC(SYSDATE) - TRUNC(v_return_date));

    CASE v_condition
        WHEN 'damaged' THEN
            v_fine := 50 + (v_days_late * 3);
        WHEN 'good' THEN
            v_fine := v_days_late * 3;
    END CASE;

    RETURN v_fine;
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Issue occurred while calculating the fine: ' || SQLERRM);
        RETURN 0;
END;
```

פונקציה 2: count_borrowed_books

מטרת הפונקציה:
לחשב את מספר הספרים שמושאלים כעת על ידי סטודנט מסוים.

הסבר הפונקציה:
הפונקציה מבצעת שאילתה לטבלת Lent ומחזירה את מספר הספרים שמצבם הוא "borrowed" עבור מזהה סטודנט שניתן כקלט. אם לא נמצאו ספרים מושאלים, היא מחזירה 0.

```oracle_plsql
CREATE OR REPLACE FUNCTION count_borrowed_books(v_student_id IN NUMBER)
RETURN NUMBER IS
    v_borrowed_count NUMBER;
BEGIN
    SELECT COUNT(*)
    INTO v_borrowed_count
    FROM Lent
    WHERE Student_id = v_student_id
      AND Status = 'borrowed';

    RETURN v_borrowed_count;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('No borrowed books found for the student.');
        RETURN 0;
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Issue occurred while counting borrowed books: ' || SQLERRM);
        RETURN 0;
END;
```

פונקציה 3: get_available_copies

מטרת הפונקציה:
להחזיר את מספר העותקים הזמינים של ספר מסוים.

הסבר הפונקציה:
הפונקציה מבצעת חישוב סכום של עותקים זמינים מתוך טבלת BookStatuses עבור מזהה ספר נתון. אם לא נמצאו רשומות עבור הספר, היא מחזירה 0. הפונקציה מאפשרת לבדוק אם יש עותקים **זמינים להשאלה**.

```oracle_plsql
CREATE OR REPLACE FUNCTION get_available_copies(v_book_id IN NUMBER)
RETURN NUMBER IS
    v_total_copies NUMBER;
BEGIN
    SELECT SUM(Number_of_copies)
    INTO v_total_copies
    FROM BookStatuses
    WHERE Book_id = v_book_id
      AND Status = 'available';

    RETURN NVL(v_total_copies, 0);
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('No available copies found for the book.');
        RETURN 0;
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Issue occurred while retrieving available copies: ' || SQLERRM);
        RETURN 0;
END;
```

פונקציה 4: has_borrowed_book

מטרת הפונקציה:
לבדוק אם סטודנט מסוים כבר השאיל ספר מסוים.

הסבר הפונקציה:
הפונקציה בודקת בטבלת Lent אם קיימת רשומה במצב "borrowed" עבור הסטודנט והספר. אם כן, היא מחזירה TRUE. אחרת, היא מחזירה FALSE.

```oracle_plsql
CREATE OR REPLACE FUNCTION has_borrowed_book(
    v_student_id IN NUMBER,
    v_book_id IN NUMBER
) RETURN BOOLEAN IS
    v_borrowed_count NUMBER := 0;
BEGIN
    SELECT COUNT(*)
    INTO v_borrowed_count
    FROM Lent
    WHERE Student_id = v_student_id
      AND Book_id = v_book_id
      AND Status = 'borrowed';

    RETURN v_borrowed_count > 0;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Student has not borrowed the specified book.');
        RETURN FALSE;
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Issue occurred while checking borrowed book status: ' || SQLERRM);
        RETURN FALSE;
END;
```

פונקציה 5: is_book_available

מטרת הפונקציה:
לבדוק אם ספר מסוים זמין להשאלה.

הסבר הפונקציה:
הפונקציה סופרת את מספר הרשומות בטבלת BookStatuses עם סטטוס "available" עבור מזהה ספר נתון. אם יש עותקים זמינים, היא מחזירה את מספרם. אם לא, היא מחזירה 0.

```oracle
CREATE OR REPLACE FUNCTION is_book_available(v_book_id IN NUMBER)
RETURN NUMBER IS
    v_available_count NUMBER;
BEGIN
    SELECT COUNT(*)
    INTO v_available_count
    FROM BookStatuses
    WHERE Book_id = v_book_id
      AND Status = 'available';

    RETURN v_available_count;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('No available records found for the book.');
        RETURN 0;
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Issue occurred while checking book availability: ' || SQLERRM);
        RETURN 0;
END;
```

פונקציה 6: has_high_debt

מטרת הפונקציה:
לבדוק אם סטודנט חרג מחוב של 200 שקלים.

הסבר הפונקציה:
הפונקציה מחשבת את סך כל הקנסות עבור סטודנט מסוים מתוך טבלת Lent. אם הסכום גדול מ-200, היא מחזירה TRUE, אחרת FALSE. זה עוזר להחליט אם הסטודנט רשאי להשאיל ספרים נוספים.

```oracle
CREATE OR REPLACE FUNCTION has_high_debt(v_student_id IN NUMBER)
RETURN BOOLEAN IS
    v_total_debt NUMBER;
BEGIN
    SELECT SUM(Fine_amount)
    INTO v_total_debt
    FROM Lent
    WHERE Student_id = v_student_id;

    RETURN NVL(v_total_debt, 0) > 200;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('No debt records found for the student.');
        RETURN FALSE;
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Issue occurred while checking the student debt: ' || SQLERRM);
        RETURN FALSE;
END;
```

פונקציה 7: lend_single_book

מטרת הפונקציה:
לבצע את כל התהליכים הדרושים להשאלת ספר יחיד לסטודנט, כולל עדכון מצב הספר וחישוב זמינות העותקים.

הסבר הפונקציה:
הפונקציה מבצעת בדיקות שונות כמו תוקף תאריך ההחזרה, מצב עותקים זמינים, ואם הסטודנט כבר השאיל את הספר. אם הכל תקין, היא מעדכנת את טבלת BookStatuses וטבלת Lent בהתאם. בנוסף, היא מטפלת בהוספת סטטוס חדש כמו "borrowed" במידת הצורך.

```oracle_plsql
CREATE OR REPLACE FUNCTION lend_single_book(
    v_book_id IN NUMBER,
    v_student_id IN NUMBER,
    v_return_due_date IN DATE
) RETURN BOOLEAN IS
    v_available_copies NUMBER;
    v_borrowed_status_exists NUMBER := 0;
BEGIN
    IF has_borrowed_book(v_student_id, v_book_id) THEN
        DBMS_OUTPUT.PUT_LINE('Student has already borrowed the book with ID: ' || v_book_id);
        RETURN FALSE;
    END IF;

    v_available_copies := get_available_copies(v_book_id);

    IF v_available_copies = 0 THEN
        DBMS_OUTPUT.PUT_LINE('No copies available for the book.');
        RETURN FALSE;
    END IF;

    IF v_available_copies = 1 THEN
        DELETE FROM BookStatuses
        WHERE Book_id = v_book_id AND Status = 'available';
    ELSE
        UPDATE BookStatuses
        SET Number_of_copies = Number_of_copies - 1
        WHERE Book_id = v_book_id AND Status = 'available';
    END IF;

    INSERT INTO Lent (Book_id, Student_id, Status, Fine_amount, Borrow_date, Return_date)
    VALUES (v_book_id, v_student_id, 'borrowed', 0, SYSDATE, v_return_due_date);

    RETURN TRUE;
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Issue occurred while lending the book: ' || SQLERRM);
        RETURN FALSE;
END;
```

פרוצדורה 1: lend_books

מטרת הפרוצדורה:
לטפל בהשאלת מספר ספרים לסטודנט בתהליך אחד.

הסבר הפרוצדורה:
הפרוצדורה מקבלת רשימת מזהי ספרים, מזהה סטודנט, ורשימת משכי השאלה. היא בודקת אם לסטודנט יש חובות גדולים מ-200 ואם סך כל ההשאלות שלו עומד במגבלה של 10 ספרים. לאחר מכן, היא מנסה להשאיל כל ספר ברשימה באמצעות הפונקציה lend_single_book. אם יש חריגה או בעיה, הספר הספציפי נדחה.

```oracle_plsql
CREATE OR REPLACE PROCEDURE lend_books(
    v_book_ids IN SYS.ODCINUMBERLIST,
    v_student_id IN NUMBER,
    v_borrow_durations IN SYS.ODCINUMBERLIST
) IS
    v_total_books_borrowed NUMBER;
    v_i NUMBER := 1;
    v_success BOOLEAN;
BEGIN
    IF has_high_debt(v_student_id) THEN
        DBMS_OUTPUT.PUT_LINE('Student has debts exceeding 200 and cannot borrow books.');
        RETURN;
    END IF;

    v_total_books_borrowed := count_borrowed_books(v_student_id);

    WHILE v_i <= v_book_ids.COUNT AND v_total_books_borrowed < 10 LOOP
        IF v_borrow_durations(v_i) > 30 THEN
            DBMS_OUTPUT.PUT_LINE('Book ID ' || v_book_ids(v_i) || ' exceeds the 30-day limit.');
        ELSE
            v_success := lend_single_book(
                v_book_id => v_book_ids(v_i),
                v_student_id => v_student_id,
                v_return_due_date => SYSDATE + v_borrow_durations(v_i)
            );

            IF v_success THEN
                DBMS_OUTPUT.PUT_LINE('Book ID ' || v_book_ids(v_i) || ' successfully lent.');
                v_total_books_borrowed := v_total_books_borrowed + 1;
            END IF;
        END IF;
        v_i := v_i + 1;
    END LOOP;

    DBMS_OUTPUT.PUT_LINE('Books lent process completed.');
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Issue occurred while lending books: ' || SQLERRM);
END;
```

בס"ד

פרוצדורה 2: return_single_book

מטרת הפרוצדורה:
לטפל בהחזרת ספר יחיד לסטודנט, כולל עדכון מצב הספר, חישוב קנס, ועדכון רשומות.

הסבר הפרוצדורה:
הפרוצדורה מבצעת מספר שלבים: בדיקת מצב הספר המושאל, עדכון העותקים בטבלת BookStatuses, חישוב הקנס באמצעות calculate_fine, ועדכון הרשומה בטבלת Lent. היא מספקת משוב על התהליך ומוסיפה עותקים חזרה לסטטוס המתאים בהתאם למצב הספר.

```oracle_plsql
CREATE OR REPLACE PROCEDURE return_single_book(
    v_book_id IN NUMBER,
    v_student_id IN NUMBER,
    v_condition IN VARCHAR2
) IS
    v_return_date DATE;
    v_fine NUMBER;
BEGIN
    -- Check if the student has borrowed the book
    IF NOT has_borrowed_book(v_student_id, v_book_id) THEN
        DBMS_OUTPUT.PUT_LINE('Student ID ' || v_student_id || ' has not borrowed Book ID ' || v_book_id || '.');
        RETURN;
    END IF;

    -- Retrieve the return date from the Lent table
    SELECT Return_date INTO v_return_date
    FROM Lent
    WHERE Book_id = v_book_id AND Student_id = v_student_id AND Status = 'borrowed';

    -- Calculate the fine for the returned book
    v_fine := calculate_fine(v_condition, v_return_date);
    DBMS_OUTPUT.PUT_LINE('Fine calculated for Book ID ' || v_book_id || ': ' || v_fine);

    -- Update the Lent table with the return information
    UPDATE Lent
    SET Status = CASE v_condition
                    WHEN 'good' THEN 'returned'
                    WHEN 'damaged' THEN 'damaged'
                    WHEN 'lost' THEN 'lost'
                 END,
        Fine_amount = v_fine,
        Return_date = SYSDATE
    WHERE Book_id = v_book_id AND Student_id = v_student_id AND Status = 'borrowed';

    -- Update BookStatuses based on the book's condition
    CASE v_condition
        WHEN 'good' THEN
            UPDATE BookStatuses
            SET Number_of_copies = Number_of_copies + 1
            WHERE Book_id = v_book_id AND Status = 'available';
        WHEN 'damaged' THEN
            UPDATE BookStatuses
            SET Number_of_copies = Number_of_copies + 1
            WHERE Book_id = v_book_id AND Status = 'damaged';
        WHEN 'lost' THEN
            UPDATE BookStatuses
            SET Number_of_copies = Number_of_copies + 1
            WHERE Book_id = v_book_id AND Status = 'lost';
        ELSE
            DBMS_OUTPUT.PUT_LINE('Invalid condition specified for the book. Update not performed.');
            RETURN;
    END CASE;

    -- Handle cases where the book's status record doesn't exist
    IF SQL%ROWCOUNT = 0 THEN
        INSERT INTO BookStatuses (Book_id, Status, Number_of_copies)
        VALUES (v_book_id, v_condition, 1);
    END IF;

    DBMS_OUTPUT.PUT_LINE('Book ID ' || v_book_id || ' successfully returned by Student ID ' || v_student_id || '.');
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Issue occurred while returning Book ID ' || v_book_id || ': ' || SQLERRM);
END;
```

פרוצדורה 3: return_books

מטרת הפרוצדורה:
לטפל בהחזרת מספר ספרים לסטודנט בתהליך אחד.

הסבר הפרוצדורה:
הפרוצדורה מקבלת רשימת מזהי ספרים ורשימת מצבים תואמת, ומבצעת את ההחזרה עבור כל ספר באמצעות הפרוצדורה return_single_book. היא מספקת משוב על כל ספר ומטפלת בחריגות אפשריות לכל ספר בנפרד, כך שהתהליך ממשיך גם אם יש בעיה בספר מסוים.

```oracle_plsql
CREATE OR REPLACE PROCEDURE return_books(
    v_student_id IN NUMBER,
    v_book_ids IN SYS.ODCINUMBERLIST,
    v_conditions IN SYS.ODCIVARCHAR2LIST
) IS
BEGIN
    -- Validate input: number of books must match the number of conditions
    IF v_book_ids.COUNT != v_conditions.COUNT THEN
        DBMS_OUTPUT.PUT_LINE('Mismatch between the number of books and conditions. Please ensure they match.');
        RETURN;
    END IF;

    -- Iterate through the list of books and their conditions
    FOR i IN 1..v_book_ids.COUNT LOOP
        BEGIN
            DBMS_OUTPUT.PUT_LINE('Processing return for Book ID: ' || v_book_ids(i) || ', Condition: ' || v_conditions(i));

            -- Call the single book return procedure
            return_single_book(v_book_ids(i), v_student_id, v_conditions(i));

            DBMS_OUTPUT.PUT_LINE('Book ID ' || v_book_ids(i) || ' successfully processed.');
        EXCEPTION
            WHEN OTHERS THEN
                DBMS_OUTPUT.PUT_LINE('Issue occurred while processing Book ID ' || v_book_ids(i) || ': ' || SQLERRM);
        END;
    END LOOP;

    DBMS_OUTPUT.PUT_LINE('All books processed for Student ID ' || v_student_id || '.');
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Issue occurred while returning books for Student ID ' || v_student_id || ': ' || SQLERRM);
END;
```

תוכנית ראשית (Main) עבור השאלת ספרים:

```
DECLARE
  v_student_id NUMBER := 271247496; -- Student ID
  v_book_ids SYS.ODCINUMBERLIST := SYS.ODCINUMBERLIST(3313, 3015, 3016); -- List of Book IDs
  v_borrow_durations SYS.ODCINUMBERLIST := SYS.ODCINUMBERLIST(15, 20, 35); -- Borrow durations in days for each book
BEGIN
  -- Call the lend_books procedure
  lend_books(
    v_book_ids => v_book_ids,
    v_student_id => v_student_id,
    v_borrow_durations => v_borrow_durations
  );
END;
```

נעשה בדיקה של השאלת ספרים עבוד הסטודנט עם תעודת הזהות 271247496.
לפני ההרצה אפשר לראות שאלו הספרים שכרגע הוא השאיל (ולא החזיר):

```
select * from lent
where student_id= 271247496 and status = 'borrowed';
```

	BOOK_ID	STUDENT_ID	STATUS	FINE_AMOUNT	BORROW_DATE	RETURN_DATE
1	3313	271247496	borrowed	0	10-11-2024...	22-11-2024 00:00:00
2	3014	271247496	borrowed	0	02-12-2024...	22-12-2024 03:18:50
3	3304	271247496	borrowed	0	02-11-2024...	23-11-2024 00:00:00

נריץ את תוכנית כדי שהסטודנט ישאיל את הספרים 3313 ל-15 ימים, 3015 ל-20 ימים ו- 3016 ל-35 ימים:

```
Student has already borrowed the book with ID: 3313
Book ID 3015 successfully lent.
Book ID 3016 exceeds the 30-day limit.
Books lent process completed.


PL/SQL procedure successfully completed.
```

אפשר לראות בהדפסה שהבקשה שלו להשאיל את ספר 3313 נדחתה מכיוון שספר זה כבר קיים אצלו בהשאלה. ספר 3015 הושאל בהצלחה. והבקשה להשאיל את ספר 3016 נדחתה מכיוון שהוא ביקש להשאיל יותר מ-30 ימים.

בדיקה של שינוי הנתונים:

	BOOK_ID	STUDENT_ID	STATUS	FINE_AMOUNT	BORROW_DATE	RETURN_DATE
1	3313	271247496	borrowed	0	10-11-2024...	22-11-2024 00:00:00
2	3014	271247496	borrowed	0	02-12-2024...	22-12-2024 03:18:50
3	3015	271247496	borrowed	0	02-12-2024...	22-12-2024 05:20:34
4	3304	271247496	borrowed	0	02-11-2024...	23-11-2024 00:00:00

ואכן, קיבלנו שספר 3015 (שהיחיד שהבקשה תקינה) התווסף לרשימה.

תוכנית ראשית (Main) עבור החזרת ספרים:

```
DECLARE
  v_student_id NUMBER := 271247496; -- Student ID
  v_book_ids SYS.ODCINUMBERLIST := SYS.ODCINUMBERLIST(3313, 3015, 3016); -- List of Book IDs to return
  v_conditions SYS.ODCIVARCHAR2LIST := SYS.ODCIVARCHAR2LIST('good', 'damaged', 'lost'); -- Conditions for each book
BEGIN
  -- Call the return_books procedure
  return_books(
    v_student_id => v_student_id,
    v_book_ids => v_book_ids,
    v_conditions => v_conditions
  );
END;
```

עבור אותו סטודנט נרצה להחזיר את הספרים 3313, 3015, 3016:

```
Processing return for Book ID: 3313, Condition: good
Fine calculated for Book ID 3313: 30
Book ID 3313 successfully returned by Student ID 271247496.
Book ID 3313 successfully processed.
Processing return for Book ID: 3015, Condition: damaged
Fine calculated for Book ID 3015: 50
Book ID 3015 successfully returned by Student ID 271247496.
Book ID 3015 successfully processed.
Processing return for Book ID: 3016, Condition: lost
Student ID 271247496 has not borrowed Book ID 3016.
Book ID 3016 successfully processed.
All books processed for Student ID 271247496.
```

קיבלנו שהספר 3313 חזר במצב טוב עם איחור של 10 ימים – כלומר עם קנס של 30 שקלים.
ספר 3015 חזר במצב של נזק אז הקנס הוא 50 שקלים.
והספר 3016 בכלל לא היה בהשאלה אצל הסטודנט.

הספרים שהוחזרו כבר לא ברשימה של הספרים שהסטודנט השאיל:

	BOOK_ID	STUDENT_ID	STATUS	FINE_AMOUNT	BORROW_DATE	RETURN_DATE
1	3014	271247496	borrowed	0	02-12-2024...	22-12-2024 03:18:50
2	3304	271247496	borrowed	0	02-11-2024...	23-11-2024 00:00:00

והספרים שהוחזרו עם נתונים תקינים

	BOOK_ID	STUDENT_ID	STATUS	FINE_AMOUNT	BORROW_DATE	RETURN_DATE
1	3015	271247496	damaged	50	02-12-2024 05:20:34	02-12-2024 05:24:05
2	3313	271247496	returned	30	10-11-2024 00:00:00	02-12-2024 05:24:05