Yash Morankar
TECOB220

Assignment No. 4

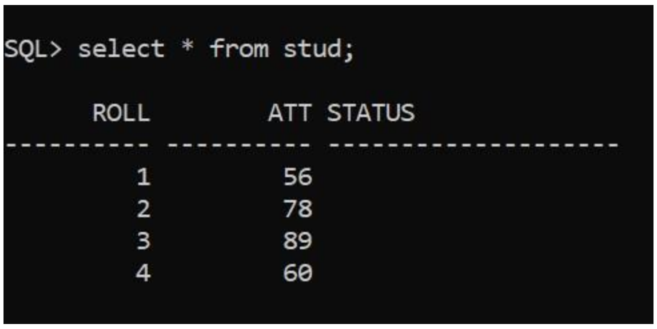
Problem Statement 1:

Consider table Stud(Roll, Att,Status)
Write a PL/SQL block for following requirement and handle the exceptions.
Roll no. of student will be entered by user. Attendance of roll no. entered by user will be checked in Stud table. If attendance is less than 75% then display the message “Term not granted” and set the status in stud table as “D”. Otherwise display message “Term granted” and set the status in stud table as “ND”

Solution:

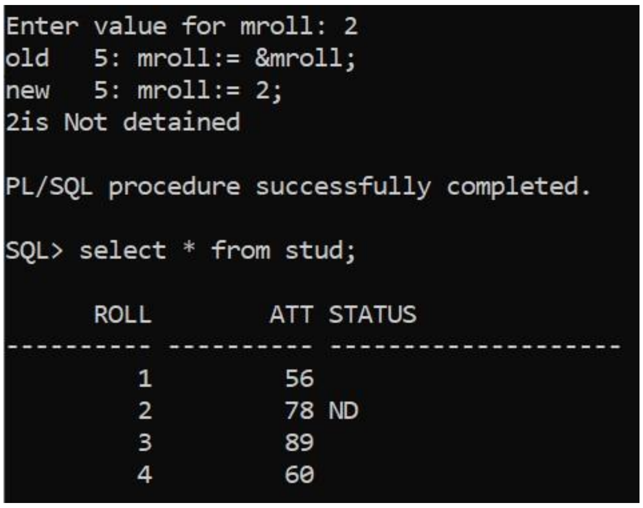
```
Declare mroll
        number(10); matt
        number(10);

Begin mroll:= &mroll;

        select att into matt from stud where roll = mroll;

        if matt<75 then dbms_output.put_line(mroll||'is
        detained');
                update stud set status='D'where roll=mroll;
        else
                dbms_output.put_line(mroll||'is Not
                detained');
                update stud set status='ND'where roll=mroll;
        end if;

        Exception when no_data_found
                then
                dbms_output.put_line(mroll||'Not found'); End;
```

Table stud:



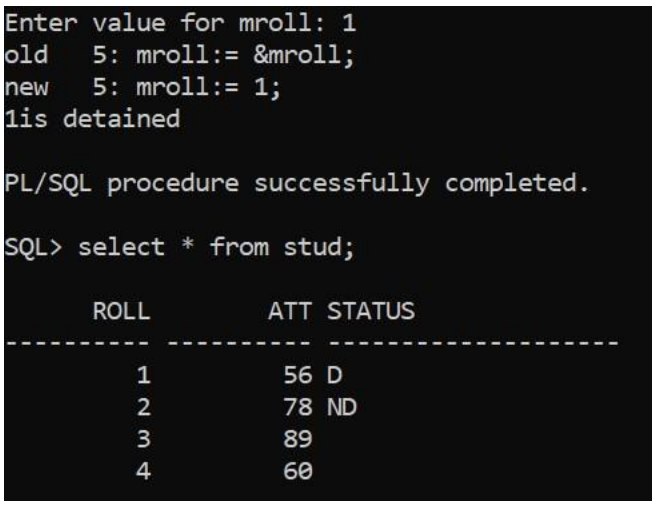
```
SQL> select * from stud;

      ROLL        ATT STATUS
---------- ---------- --------------------
         1         56
         2         78
         3         89
         4         60
```

Attendance > 75:



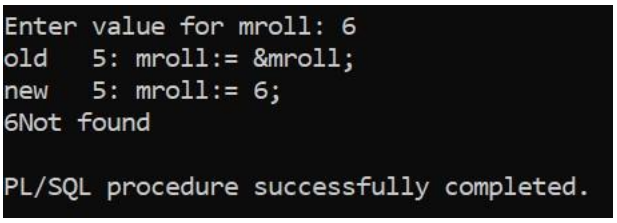
```
Enter value for mroll: 2
old   5: mroll:= &mroll;
new   5: mroll:= 2;
2is Not detained

PL/SQL procedure successfully completed.

SQL> select * from stud;

      ROLL        ATT STATUS
---------- ---------- --------------------
         1         56
         2         78 ND
         3         89
         4         60
```

Attendance < 75:



```
Enter value for mroll: 1
old   5: mroll:= &mroll;
new   5: mroll:= 1;
1is detained

PL/SQL procedure successfully completed.

SQL> select * from stud;

      ROLL        ATT STATUS
---------- ---------- --------------------
         1         56 D
         2         78 ND
         3         89
         4         60
```

Roll_no not found:



```
Enter value for mroll: 6
old   5: mroll:= &mroll;
new   5: mroll:= 6;
6Not found

PL/SQL procedure successfully completed.
```

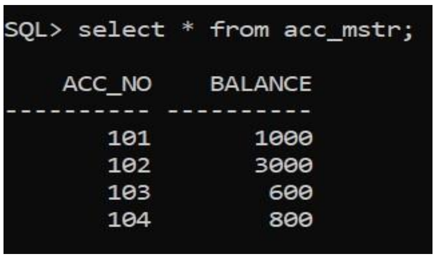
Problem Statement 2:

Write a PL/SQL block for following requirement using user defined exception handling.
The account_master table records the current balance for an account, which is updated whenever, any deposits or withdrawals takes place. If the withdrawal attempted is more than the current balance held in the account. The user defined exception is raised, displaying an appropriate message. Write a PL/SQL block for above requirement using user defined exception handling.

Solution:

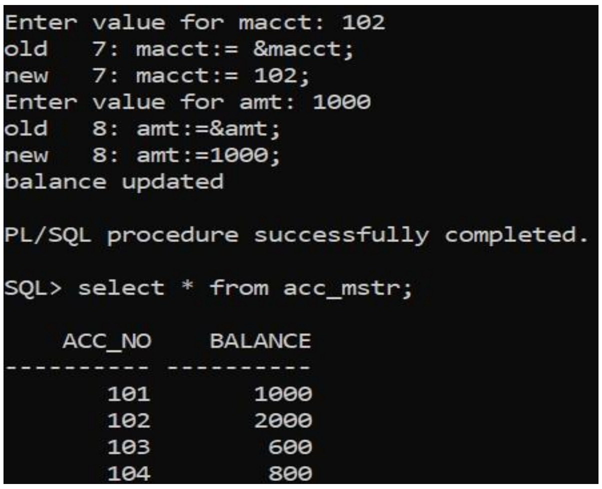
```
Declare macct
        number(20); mbal
        number(20); amt
        number(20);
        insufficient_bal exception;

Begin macct:= &macct;
        amt:=&amt;
        select balance into mbal from acc_mstr where acc_no = macct;
        if amt>mbal
                then raise insufficient_bal;
        else
                update acc_mstr set balance=mbal-amt where
                acc_no=macct; dbms_output.put_line(' balance updated');
        end if;
        Exception when insufficient_bal
                then
                dbms_output.put_line('Insufficient balance'); End;
```

Table acc_mstr :



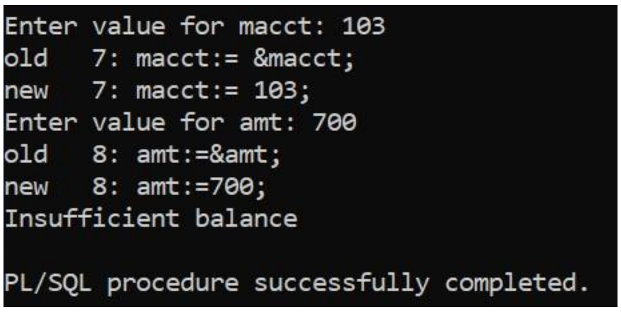
```
SQL> select * from acc_mstr;

    ACC_NO    BALANCE
---------- ----------
       101       1000
       102       3000
       103        600
       104        800
```

Sufficient balance :



```
Enter value for macct: 102
old   7: macct:= &macct;
new   7: macct:= 102;
Enter value for amt: 1000
old   8: amt:=&amt;
new   8: amt:=1000;
balance updated

PL/SQL procedure successfully completed.

SQL> select * from acc_mstr;

    ACC_NO    BALANCE
---------- ----------
       101       1000
       102       2000
       103        600
       104        800
```

Insufficient balance :



```
Enter value for macct: 103
old   7: macct:= &macct;
new   7: macct:= 103;
Enter value for amt: 700
old   8: amt:=&amt;
new   8: amt:=700;
Insufficient balance

PL/SQL procedure successfully completed.
```

Problem Statement 3:

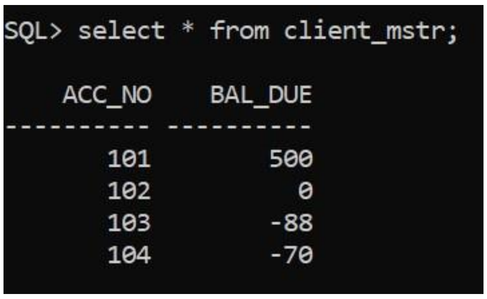
Write an SQL code block these raise a user defined exception where business rule is voilated. BR for client_master table specifies when the value of bal_due field is less than 0 handle the exception.

Solution:

```
Declare macct number(20);
        mbal number(20);
        rule_violated exception;

Begin macct:= &macct;
        select bal_due into mbal from client_mstr where acc_no =
        macct;
        if mbal<0
                then raise rule_violated;
        else
                dbms_output.put_line(' Sufficient
                balance');
        end if;
        Exception when rule_violated
                then
```

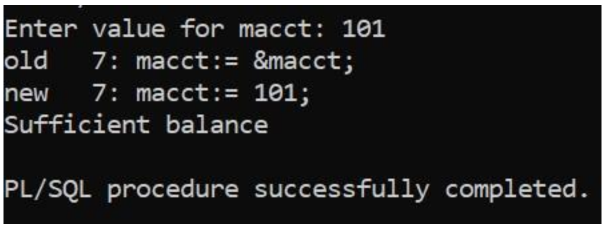
```
                    dbms_output.put_line('Rule Violated');
        End;
```

Table client_mstr :



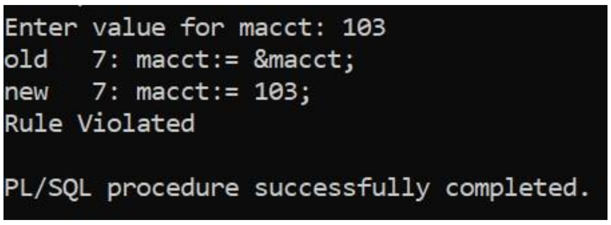
```
SQL> select * from client_mstr;

    ACC_NO    BAL_DUE
---------- ----------
       101        500
       102          0
       103        -88
       104        -70
```

Sufficient Balance :



```
Enter value for macct: 101
old   7: macct:= &macct;
new   7: macct:= 101;
Sufficient balance

PL/SQL procedure successfully completed.
```

Rule violated :



```
Enter value for macct: 103
old   7: macct:= &macct;
new   7: macct:= 103;
Rule Violated

PL/SQL procedure successfully completed.
```

Problem Statement 4:

Consider Tables:
1. Borrower(Roll_no, Name, DateofIssue, NameofBook, Status)
2. Fine(Roll_no,Date,Amt)

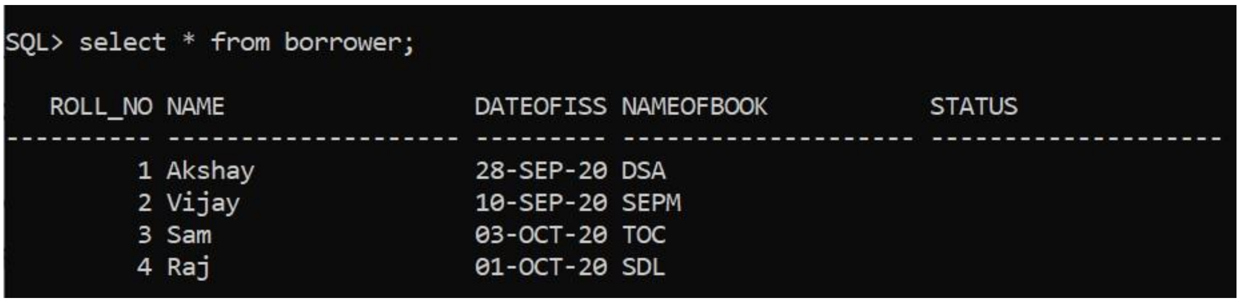
- Accept roll_no & name of book from user. Check the number of
- days (from date of issue), if days are between 15 to 30 then fine amount will be Rs 5per day.
- If no. of days>30, per day fine will be Rs 50 per day & for days less than 30, Rs. 5 per day.
- After submitting the book, status will change from I to R.
- If condition of fine is true, then details will be stored into fine table.

Also handles the exception by named exception handler or user define exception handler.

Solution:

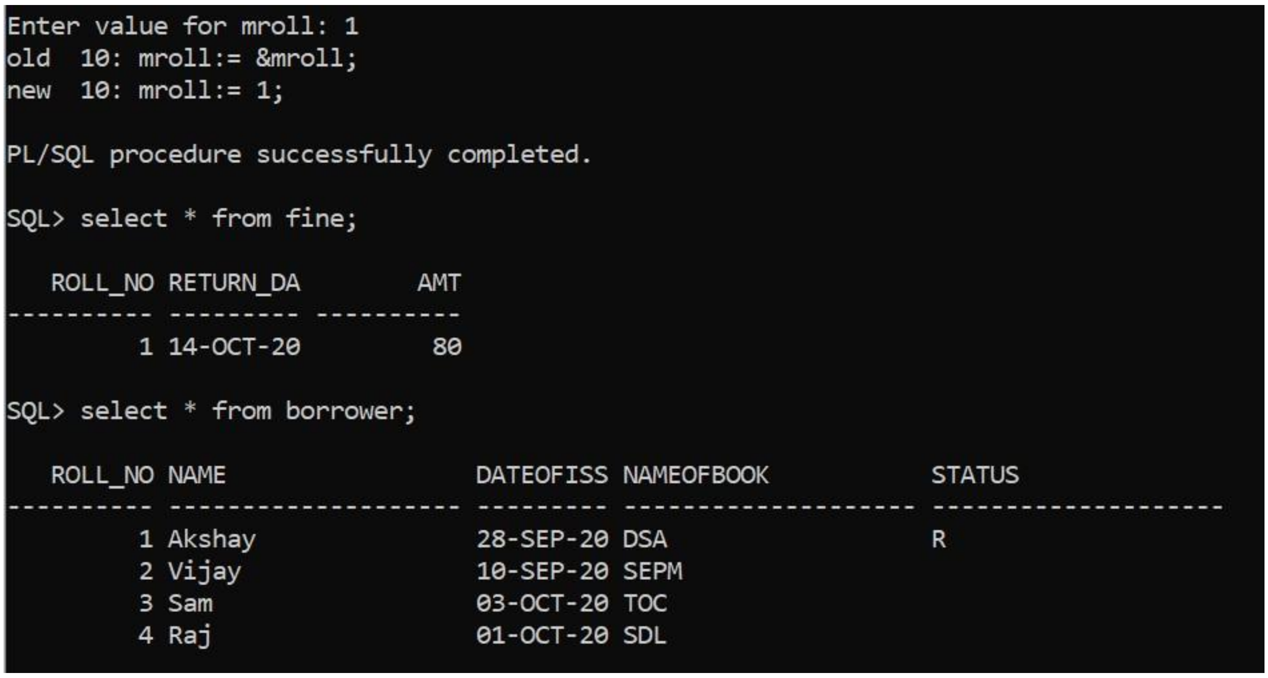
```
Declare mroll
            number(20);
            mdate date;
            sdate date; mdays
            number(20); mfine
            number(20);
            no_fine exception;

Begin mroll:= &mroll;
            select DateofIssue into mdate from borrower where roll_no = mroll;
            select sysdate into sdate from dual; mdays:=to_date(sdate)-
            to_date(mdate);
            if mdays>=15 and mdays<=30 then mfine:=mdays*5;
                        insert into fine values(mroll,sdate,mfine); update
                        borrower set status='R' where roll_no=mroll;
            elsif mdays>30 then
                        mfine:=mdays*50;
                        insert into fine values(mroll,sdate,mfine); update
                        borrower set status='R' where roll_no=mroll;
            elsif mdays<15 then update borrower set status='R' where
            roll_no=mroll; raise no_fine; end if;
                        Exception when
                        no_fine then
                        dbms_output.put_line('No fine'); End;
```

Table borrower :

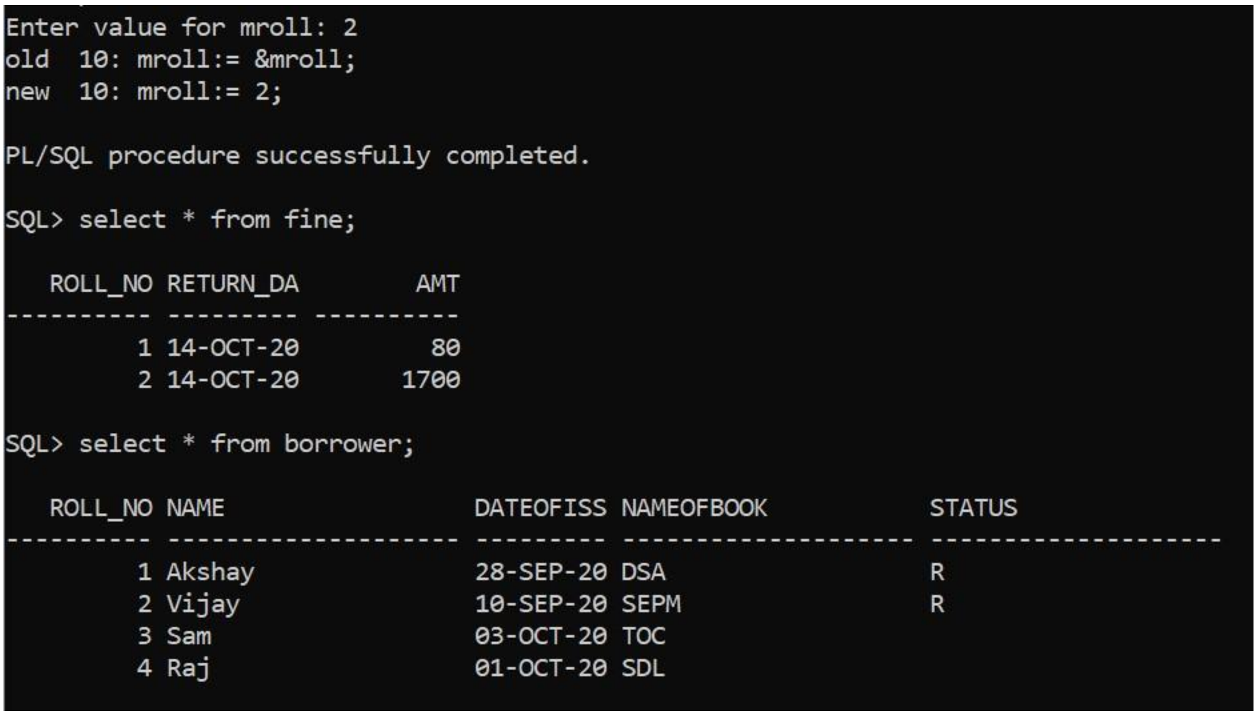


```
SQL> select * from borrower;

   ROLL_NO NAME                 DATEOFISS NAMEOFBOOK           STATUS
---------- -------------------- --------- -------------------- --------------------
         1 Akshay               28-SEP-20 DSA
         2 Vijay                10-SEP-20 SEPM
         3 Sam                  03-OCT-20 TOC
         4 Raj                  01-OCT-20 SDL
```

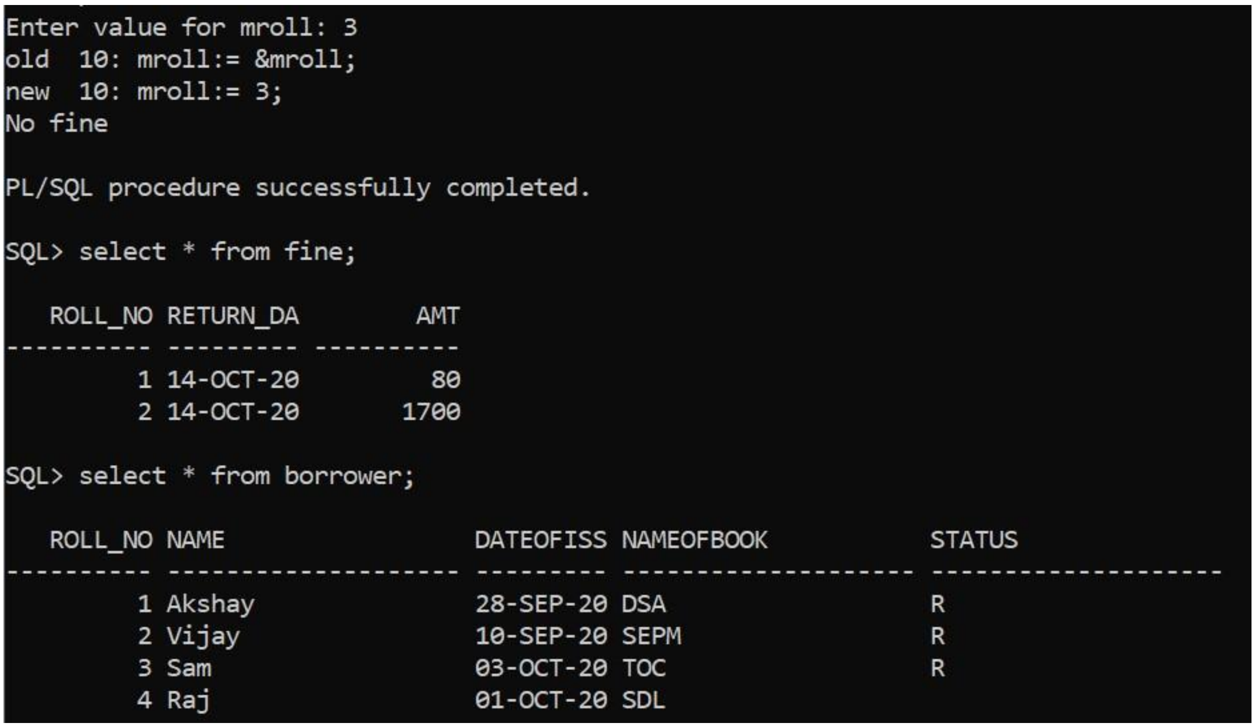
15 >= Days <=30 :


```
Enter value for mroll: 1
old  10: mroll:= &mroll;
new  10: mroll:= 1;

PL/SQL procedure successfully completed.

SQL> select * from fine;

   ROLL_NO RETURN_DA        AMT
---------- --------- ----------
         1 14-OCT-20         80

SQL> select * from borrower;

   ROLL_NO NAME                 DATEOFISS NAMEOFBOOK           STATUS
---------- -------------------- --------- -------------------- --------------------
         1 Akshay               28-SEP-20 DSA                  R
         2 Vijay                10-SEP-20 SEPM
         3 Sam                  03-OCT-20 TOC
         4 Raj                  01-OCT-20 SDL
```

Days > 30 :


```
Enter value for mroll: 2
old  10: mroll:= &mroll;
new  10: mroll:= 2;

PL/SQL procedure successfully completed.

SQL> select * from fine;

   ROLL_NO RETURN_DA        AMT
---------- --------- ----------
         1 14-OCT-20         80
         2 14-OCT-20       1700

SQL> select * from borrower;

   ROLL_NO NAME                 DATEOFISS NAMEOFBOOK           STATUS
---------- -------------------- --------- -------------------- --------------------
         1 Akshay               28-SEP-20 DSA                  R
         2 Vijay                10-SEP-20 SEPM                 R
         3 Sam                  03-OCT-20 TOC
         4 Raj                  01-OCT-20 SDL
```

Days < 15 :



```
Enter value for mroll: 3
old  10: mroll:= &mroll;
new  10: mroll:= 3;
No fine

PL/SQL procedure successfully completed.

SQL> select * from fine;

   ROLL_NO RETURN_DA        AMT
---------- --------- ----------
         1 14-OCT-20         80
         2 14-OCT-20       1700

SQL> select * from borrower;

   ROLL_NO NAME                 DATEOFISS NAMEOFBOOK           STATUS
---------- -------------------- --------- -------------------- --------------------
         1 Akshay               28-SEP-20 DSA                  R
         2 Vijay                10-SEP-20 SEPM                 R
         3 Sam                  03-OCT-20 TOC                  R
         4 Raj                  01-OCT-20 SDL
```