**Trigger**

1. **Students can borrow up to 5 books and staff can borrow up to 10 books**

CREATE OR REPLACE TRIGGER Stu\_Five

BEFORE INSERT ON Borrow

FOR EACH ROW

DECLARE

number1 int;

number2 number(4);

BEGIN

SELECT MemberID INTO number2 FROM Memb WHERE :NEW.MemberID=MemberID ;

SELECT COUNT(\*) INTO number1 FROM Borrow WHERE Borrow.MemberID = :NEW.MemberID AND Borrow.BStatus = 'Borrowing';

IF number2=:NEW.MemberID and number1=5 then

raise\_application\_error (-20004,'The maximum number of loan has been reached for a student!');

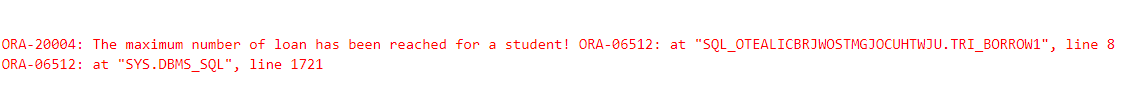
END IF;

END;

**Test Data**

SELECT \* FROM Borrow;

INSERT INTO Borrow VALUES(1004,1112,'Borrowing', TO\_DATE('20221209','yyyymmdd'), NULL);



CREATE OR REPLACE TRIGGER Sta\_Ten

BEFORE INSERT ON Borrow

FOR EACH ROW

DECLARE

number3 int;

number4 number(4);

BEGIN

SELECT MemberID INTO number4 FROM Memb WHERE :NEW.MemberID=MemberID;

SELECT COUNT(\*) INTO number3 FROM Borrow WHERE Borrow.memberID = :NEW.MemberID AND Borrow.BStatus = 'Borrowing';

IF number4=:NEW.MemberID and number3=10 then

raise\_application\_error (-20004,'The maximum number of loan has been reached for a staff!');

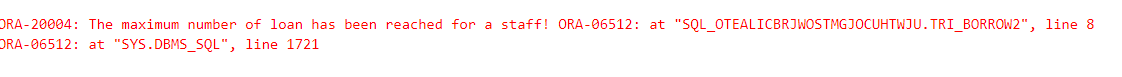
END IF;

END;

**Test Data**

SELECT \* FROM Borrow;

INSERT INTO Borrow VALUES(2001,1112,'Borrowing', TO\_DATE('20221209','yyyymmdd'), NULL);



1. **Judge whether the amount of resource is enough. If it is not enough, report an error and lend it out when it is enough**

CREATE OR REPLACE TRIGGER Lend\_Resource

BEFORE INSERT ON Borrow

FOR EACH ROW

DECLARE amount int;

BEGIN

SELECT amount INTO amount FROM Reso WHERE :NEW.ResourceID=ResourceID;

IF amount=0 then

raise\_application\_error (-20003,'No inventory!');

ELSIF amount > 0 then

UPDATE Reso SET amount = amount - 1 WHERE :New.ResourceID = ResourceID;

END IF;

END;

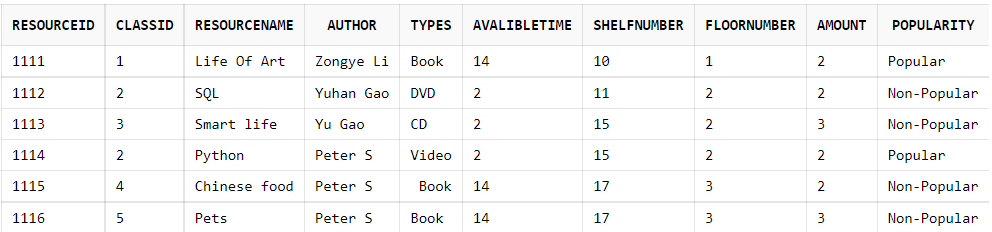
**Test Data**

SELECT \* FROM Reso;

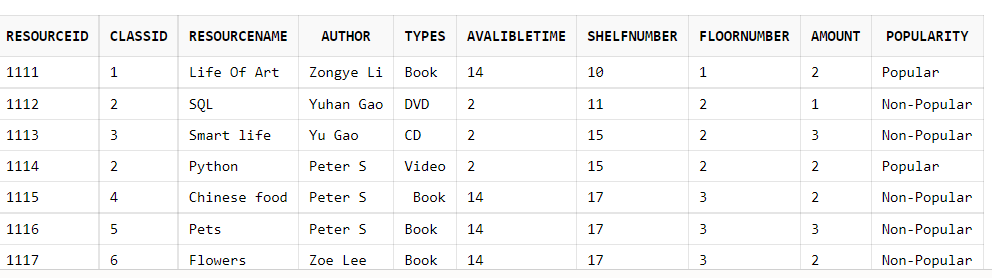
SELECT \* FROM Borrow;

INSERT INTO Borrow VALUES(2002,1112,'Borrowing', TO\_DATE('20221209','yyyymmdd'), NULL);

**Befor Insert**



**After Insert**

****

1. **Update the penalty for each person who borrows a resource**

CREATE OR REPLACE TRIGGER Update\_Penalty

AFTER UPDATE OF ReturnDate ON Borrow

FOR EACH ROW

DECLARE

a\_time int;

BEGIN

SELECT AvalibleTime INTO a\_time FROM Reso WHERE ResourceID=:NEW.ResourceID;

IF TRUNC(:NEW.ReturnDate) - TRUNC(:NEW.StartDate)>a\_time Then

UPDATE Loan set LAmount=(TRUNC(:NEW.ReturnDate) - TRUNC(:NEW.StartDate))-a\_time WHERE :NEW.MemberID=MemberID AND :NEW.ResourceID=ResourceID AND LStatus='Unpaid';

UPDATE Loan set LStatus='Paid' WHERE :NEW.MemberID=MemberID AND :NEW.ResourceID=ResourceID and LStatus='Unpaid';

END IF;

END;

**Test Data**

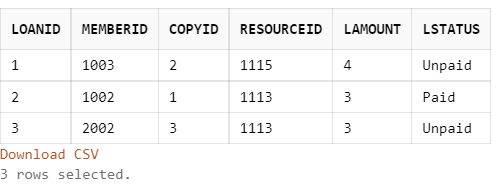
SELECT \* FROM Loan;

SELECT \* FROM Borrow;

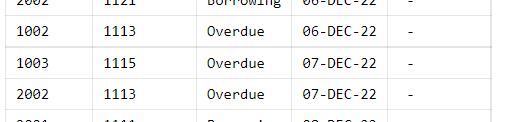
UPDATE Borrow SET ReturnDate = TO\_DATE('20221226','yyyymmdd') WHERE MemberID = 1003 AND ResourceID = 1115;

**Before**

**Loan**

****

**Borrow**

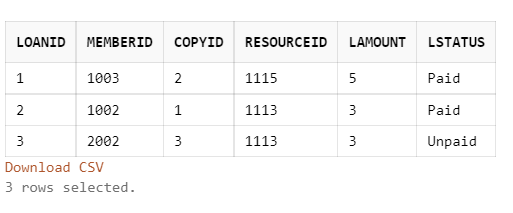
****

**Resource**

****

**After**

**Loan**

****

1. **The blacklist (all the resources a person has not paid back, add up to a total of 10 pounds in arrears)**

**You can keep borrowing books after you pay the fine**

CREATE OR REPLACE TRIGGER Not\_Allowed

BEFORE INSERT ON Borrow

FOR EACH ROW

DECLARE

total int;

BEGIN

SELECT SUM(LAmount) INTO total FROM Loan WHERE :NEW.MemberID=MemberID AND LStatus='Unpaid';

IF total>=10 THEN

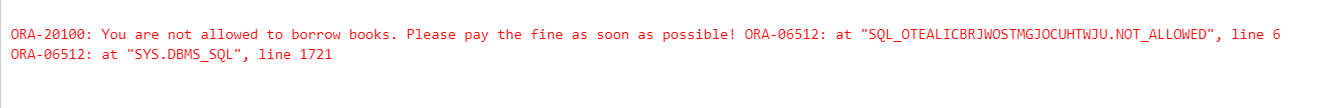
raise\_application\_error (-20100,'You are not allowed to borrow books. Please pay the fine as soon as possible!');

END IF;

END;

**Test Data**

INSERT INTO Borrow VALUES(2002,1111,'Borrowing',TO\_DATE('20221204','yyyymmdd'),NULL);



1. **When the user returns the book, the amount of the resource that can be borrowed +1**

CREATE OR REPLACE TRIGGER update\_Reso

AFTER UPDATE ON Borrow

FOR EACH ROW

DECLARE amount INT;

BEGIN

SELECT amount INTO amount FROM Reso WHERE :NEW.ResourceID = ResourceID;

CASE

WHEN UPDATING THEN

UPDATE Reso SET amount = amount + 1 WHERE :NEW.BStatus = 'Returned' AND :NEW.ResourceID = ResourceID;

END CASE;

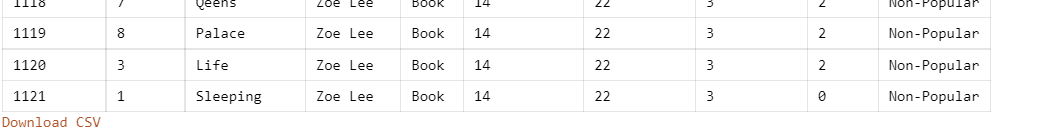
END;

**Test Data**

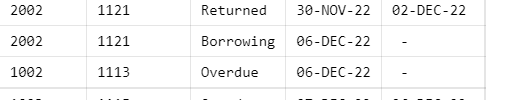
UPDATE Borrow SET BStatus = 'Returned', ReturnDate = TO\_DATE('20221207','yyyymmdd') WHERE MemberID = 2002 AND ResourceID = 1121 AND StartDate = TO\_DATE('20221206','yyyymmdd');

**Before**

Resource

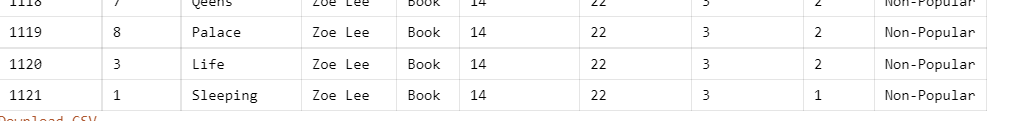


Borrow

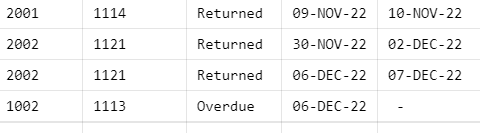


**After**

Resource



Borrow



**6.**

CREATE OR REPLACE TRIGGER update\_popularity

BEFORE INSERT ON Borrow

FOR EACH ROW

DECLARE Amount\_Borrow INT;

BEGIN

SELECT COUNT(\*) INTO Amount\_Borrow FROM Borrow WHERE :NEW.ResourceID = ResourceID;

IF Amount\_Borrow >= 4 THEN

UPDATE Reso SET Popularity = 'Popular' WHERE :NEW.ResourceID = ResourceID;

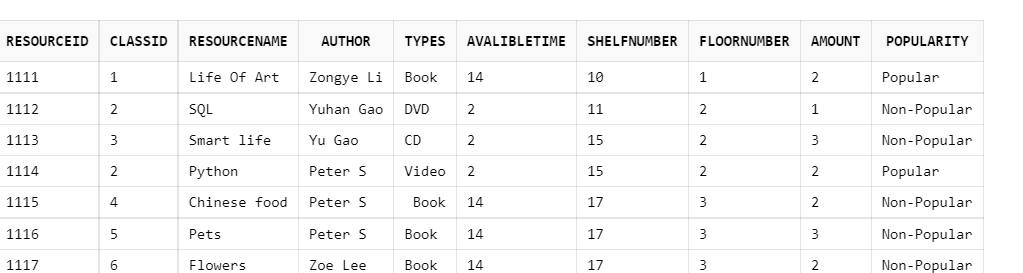
END IF;

END;

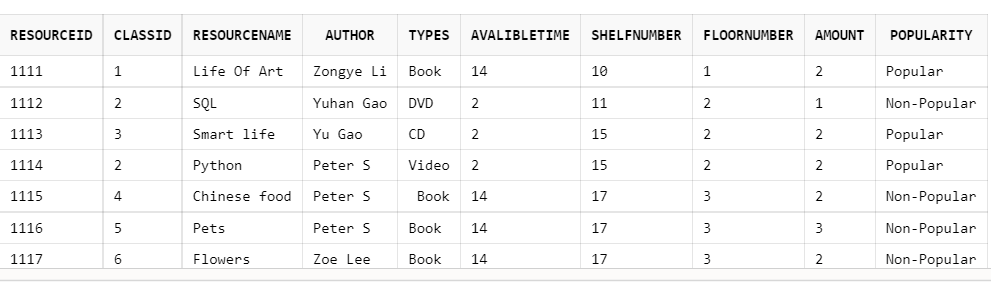
**Test Data**

INSERT INTO Borrow VALUES(1005,1113,'Borrowing', TO\_DATE('20221212','yyyymmdd'),NULL)

**Before**

****

**After**

****