**UPDATE BRANCH ADDRESS – NOT DEPLOYED**

DELIMITER $$

CREATE PROCEDURE update\_branch(IN bid INT,IN bname VARCHAR(100),IN baddress VARCHAR(100),

OUT message VARCHAR(100))

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

SELECT 'Error with procedure' INTO message;

END;

IF (bname = 'n/a' OR bname = 'N/A') THEN

UPDATE tbl\_library\_branch

SET branchAddress = baddress

WHERE branchId = bid;

SELECT CONCAT('The Branch address for Branch ID: ',bid,' has been updated to ',baddress)

INTO message;

ELSEIF (baddress = 'n/a' OR baddress = 'N/A') THEN

UPDATE tbl\_library\_branch

SET branchName = bname

WHERE branchId = bid;

SELECT CONCAT('The Branch Name for Branch ID: ',bid,' has been updated to ',bname)

INTO message;

ELSE

UPDATE tbl\_library\_branch

SET branchAddress = baddress AND branchName = bname

WHERE branchId = bid;

SELECT CONCAT('The Branch Name for Branch ID: ',bid,' has been updated to ',

bname,' with address ',baddress)

INTO message;

END IF;

COMMIT;

END $$

DELIMITER ;

**STORED PROCEDURE TO VERIFY CARD NUMBER – NOT DEPLOYED**

DELIMITER $$

CREATE PROCEDURE verify\_cardno(cardnumber INT, OUT message VARCHAR(100), OUT flag BOOLEAN)

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

SELECT 'Error with procedure' INTO message;

END;

SELECT count(cardNo) INTO flag

FROM tbl\_borrower

WHERE cardNo = cardNumber;

COMMIT;

END $$

DELIMITER ;

**PROCEDURE FOR BOOK CHECKOUT – NOT DEPLOYED**

DELIMITER $$

-- PROCEDURE to call Borrower and store the details in the table (cardNo needs to be unique)

CREATE PROCEDURE Loan\_checkout(bookId1 INT, branchId1 INT, cardNo1 INT, dateOut1 datetime, dueDate1 datetime, returnDate1 datetime)

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

SELECT 'Error with procedure' INTO message;

END;

BEGIN

DECLARE copies INT;

SET copies = ( SELECT noOfCopies

FROM tbl\_book\_copies

WHERE bookId1 = bookId AND branchId1 = branchId);

IF (copies > 0) THEN

INSERT INTO library.tbl\_book\_loans (bookId, branchId, cardNo, dateOut, dueDate, returnDate)

VALUES (bookId1, branchId1, cardNo1, dateOut1, dueDate1, returnDate1);

ELSE

Select concat('You don','t have any more copies',copies,' of this book in this branch ',branchId1) AS CHECKOUT;

END IF;

COMMIT;

END$$

DELIMITER ;

**PROCEDURE to call Book return – NOT USED**

DELIMITER $$

-- PROCEDURE to call Book return and update the return date for the user returning the book

CREATE PROCEDURE Loan\_return(bookId1 INT, cardNo1 INT, returndate1 datetime)

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

SELECT 'Error with procedure' INTO message;

END;

DECLARE result int;

UPDATE tbl\_book\_loans

SET returnDate = returndate1

WHERE bookId1 = bookID AND cardNo1 = cardNo;

SET result = (SELECT datediff(returnDate1,(SELECT dueDate

FROM tbl\_book\_loans

WHERE cardNo = cardNo1 AND bookId = bookId1)));

IF (result>0) THEN

select concat("Your penalty is :", result\*0.25, '$');

else

SELECT 'No Penalty';

END IF;

COMMIT;

END$$

DELIMITER ;

**PRINT BOOKS CHECKED OUT BY THE BORROWER – NOT DEPLOYED**

DELIMITER $$

CREATE PROCEDURE borrowed\_list(bid INT, cardnumber INT, OUT message VARCHAR(100))

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

SELECT 'Error with procedure' INTO message;

END;

BEGIN TRANSCATION

SELECT title AS 'Title', dateOut AS 'Date Checked Out', dueDate AS 'Date Due'

FROM tbl\_book\_loans

JOIN tbl\_book ON tbl\_book.bookId = tbl\_book\_loans.bookId

WHERE bookId = bid AND cardNo = cardnumber

ORDER BY title;

COMMIT;

END $$

DELIMITER ;

**PROCEDURE TO ADD BOOK LOAN - USED**

delimiter $$

CREATE PROCEDURE add\_loan(in bId INT, in brId INT, in cardnumber INT,

out mssg varchar(50))

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

SELECT 'SQLException encountered. Procedure cancelled.' INTO mssg;

END;

-- check that the branch with branchId = oldId exists

IF((SELECT COUNT(branchId) FROM tbl\_library\_branch WHERE branchId = brId) > 0

AND (SELECT COUNT(bookId) FROM tbl\_book WHERE bookId = bId) > 0

AND (SELECT COUNT(cardNo) FROM tbl\_borrower WHERE cardNo = cardnumber) > 0 ) THEN

-- check if user entered a new name and update table if neccessary

INSERT INTO tbl\_book\_loans(bookId, branchId, cardNo, dateOut, dueDate) VALUES (bId, brId, cardnumber, now(),

date\_add(now(), INTERVAL 7 DAY));

SELECT 'Loan successfully added/updated in the table' INTO mssg;

ELSE

SELECT 'Branch already exists in the table' INTO mssg;

END IF;

COMMIT;

END $$

DELIMITER ;

**PROCEDURE TO ADD BRANCH FOR ADMIN - USED**

delimiter $$

CREATE PROCEDURE add\_branch(in bId INT, in branchName VARCHAR(45), in branchAddress VARCHAR(100),

out mssg varchar(50))

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

SELECT 'SQLException encountered. Procedure cancelled.' INTO mssg;

END;

-- check that the branch with branchId = oldId exists

IF((SELECT COUNT(branchId) FROM tbl\_library\_branch WHERE branchId = bId) < 1 ) THEN

-- check if user entered a new name and update table if neccessary

IF (branchName <> 'n/a' AND branchName <> 'N/A') THEN

INSERT INTO tbl\_library\_branch(branchId, branchName) VALUES (bId, branchName);

ELSEIF (branchAddress <> 'n/a' AND branchAddress <> 'N/A') THEN

INSERT INTO tbl\_library\_branch(branchId, branchAddress) VALUES (bId, branchAddress);

ELSE

INSERT INTO tbl\_library\_branch VALUES (bId, branchName, branchAddress);

END IF;

SELECT 'Branch successfully added in the table' INTO mssg;

ELSE

SELECT 'Branch already exists in the table' INTO mssg;

END IF;

COMMIT;

END $$

DELIMITER ;

**PROCEDURE TO ADD BORROWER FOR ADMIN - USED**

CREATE PROCEDURE add\_borrower(in cId INT, in bName VARCHAR(45), in bAddress VARCHAR(100), bphone VARCHAR(20),out mssg varchar(50))

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

SELECT 'SQLException encountered. Procedure cancelled.' INTO mssg;

END;

start transcation

-- check that the branch with branchId = oldId exists

IF((SELECT COUNT(cardNo) FROM tbl\_borrower WHERE cardNo = cId) < 1) THEN

-- check if user entered a new name and update table if neccessary

IF (bName='n/a'AND bName='N/A' AND bAddress='N/A' AND bAddress='n/a' AND bphone = 'n/a' AND bphone = 'N/A' ) THEN

INSERT INTO tbl\_borrower(cardNo) VALUES (cId);

ELSEIF (bName = 'n/a' AND bName = 'N/A' AND bphone = 'n/a' AND bphone = 'N/A') THEN

INSERT INTO tbl\_borrower(cardNo, address) VALUES (cId, bAddress);

ELSEIF (bAddress='n/a' AND bAddress='N/A' AND bphone = 'n/a' AND bphone = 'N/A') THEN

INSERT INTO tbl\_borrower(cardNo, tbl\_borrower.name) VALUES (cId, bName);

ELSEIF (bphone = 'n/a' AND bphone = 'N/A') THEN

INSERT INTO tbl\_borrower(cardNo, tbl\_borrower.name, address) VALUES (cId, bName, bAddress);

ELSEIF (bAddress='n/a' AND bAddress='N/A' AND bName ='n/a' AND bName='N/A') THEN

INSERT INTO tbl\_borrower(cardNo, phone) VALUES (cId, bphone);

ELSEIF (bName='n/a'AND bName='N/A') THEN

INSERT INTO tbl\_borrower(cardNo, address, phone) VALUES (cId, bAddress, bphone);

SELECT 'Borrower successfully added to the table' INTO mssg;

ELSEIF (bAddress='n/a' AND bAddress='N/A') THEN

INSERT INTO tbl\_borrower(cardNo, tbl\_borrower.name, phone) VALUES (cId, bName,bphone);

ELSE

INSERT INTO tbl\_borrower VALUES (cId, bName, bAddress, bphone);

SELECT 'Borrower successfully added in the table' INTO mssg;

END IF;

ELSE

SELECT 'Borrower already exists in the table' INTO mssg;

END IF;

COMMIT;

END $$

DELIMITER ;

**PROCEDURE TO ADD BOOK FOR ADMIN - USED**

CREATE DEFINER=`root`@`localhost` PROCEDURE `add\_book`(in bId INT, in bName varchar(45), in pId INT, in pName VARCHAR(45), in aName VARCHAR(45), in aId INT, in gId INT, in branchId INT, in copies INT, out mssg varchar(50))

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

SELECT 'SQLException encountered. Procedure cancelled.' INTO mssg;

END;

-- check that the book with bookId = oldId exists

IF((SELECT COUNT(bookId) FROM tbl\_book WHERE bookId = bId) < 1 ) THEN

-- check if user entered a new name and update table if neccessary

IF (bName <> 'n/a' AND bName <> 'N/A' AND aName <> 'n/a' AND aName <> 'N/A'

AND pName <> 'n/a' AND pName <> 'N/A') THEN

IF NOT EXISTS (SELECT 1 FROM tbl\_publisher WHERE publisherId = pId) THEN

INSERT INTO tbl\_publisher(publisherId, publisherName) VALUES (pId, pName);

END IF;

IF NOT EXISTS (SELECT 1 FROM tbl\_author WHERE authorId = aId) THEN

INSERT INTO tbl\_author VALUES (aId, aName);

END IF;

INSERT INTO tbl\_book VALUES (bId, bName, pId);

INSERT INTO tbl\_book\_genres VALUES (gId, bId);

INSERT INTO tbl\_book\_authors VALUES(bId, aId);

INSERT INTO tbl\_book\_copies VALUES(bId, branchId, copies);

SELECT 'Book successfully added in the table' INTO mssg;

ELSE

SELECT 'Publishher/Author already exists in the table' INTO mssg;

END IF;

ELSE

SELECT 'Book already exists in the table' INTO mssg;

END IF;

COMMIT;

END