**PROCEDURE 1 – Ensure all payments for accounts with at least ‘minUsers’ are at least for amount ‘minAmount’**

**PROCEDURE CODE**

CREATE OR REPLACE FUNCTION enforceMinPayment(minUsers INT, minAmount INT)

RETURNS void AS $$

DECLARE currentPid INT;

DECLARE currentAmount INT;

DECLARE cursor CURSOR

FOR SELECT pid, amount

FROM payment

WHERE accid IN (

SELECT a.accid

FROM account a, accountuser au

WHERE a.accid=au.accid

GROUP BY a.accid

HAVING COUNT(\*) >= minUsers);

BEGIN

OPEN cursor;

FETCH NEXT FROM cursor INTO currentPid, currentAmount;

WHILE(FOUND) LOOP

IF (currentAmount < minAmount) THEN

UPDATE payment SET amount = minAmount WHERE pid=currentPid;

END IF;

FETCH NEXT FROM cursor INTO currentPid, currentAmount;

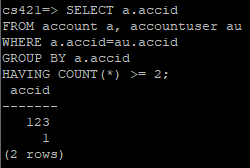
END LOOP;

CLOSE cursor;

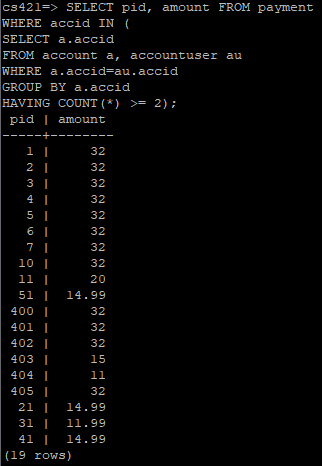
END; $$

LANGUAGE plpgsql;

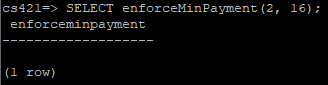
**ACCOUNTS WITH 2+ Users**

****

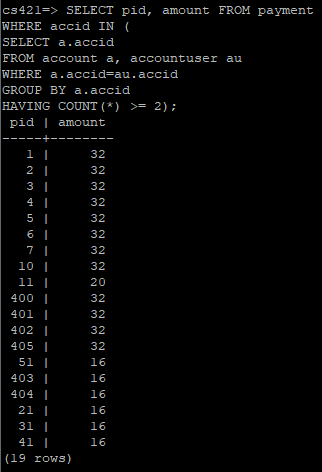
**PAYMENTS BEFORE**



**CALLING PROCEDURE**



**PAYMENTS AFTER**



**PROCEDURE 2 – Make all media available in region ‘fromRegion’ available in region ‘toRegion’**

**PROCEDURE CODE**

CREATE OR REPLACE FUNCTION copyMedia(fromRegion VARCHAR(30), toRegion VARCHAR(30))

RETURNS void AS $$

DECLARE currentMid INT;

DECLARE cursor CURSOR

FOR SELECT mid

FROM available\_in

WHERE regname=fromRegion;

BEGIN

OPEN cursor;

FETCH NEXT FROM cursor INTO currentMid;

WHILE(FOUND) LOOP

INSERT INTO available\_in VALUES (currentMid, toRegion);

FETCH NEXT FROM cursor INTO currentMid;

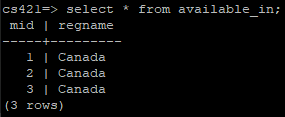
END LOOP;

CLOSE cursor;

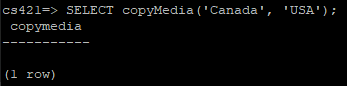
END; $$

LANGUAGE plpgsql;

**AVAILABLE\_IN BEFORE**



**CALLING PROCEDURE**



**AVAILABLE\_IN AFTER**

