**Sheet No : 5**

**Assignment No : 8**

**Write a pl/sql block for the following:**

1. **Insert data into a table containing two attributes namely radius & circumference of circles. You may get different values of radius either from keyboard or you may generate different values.**

**Creation of table:**

create table circle (

radius number,

circumference number

);

**Pl/sql block to insert values into table circle:**

declare

z number;

begin

for i in 1..50 loop

z := 100\*dbms\_random.value();

insert into circle values(z,2\*3.14\*z);

end loop;

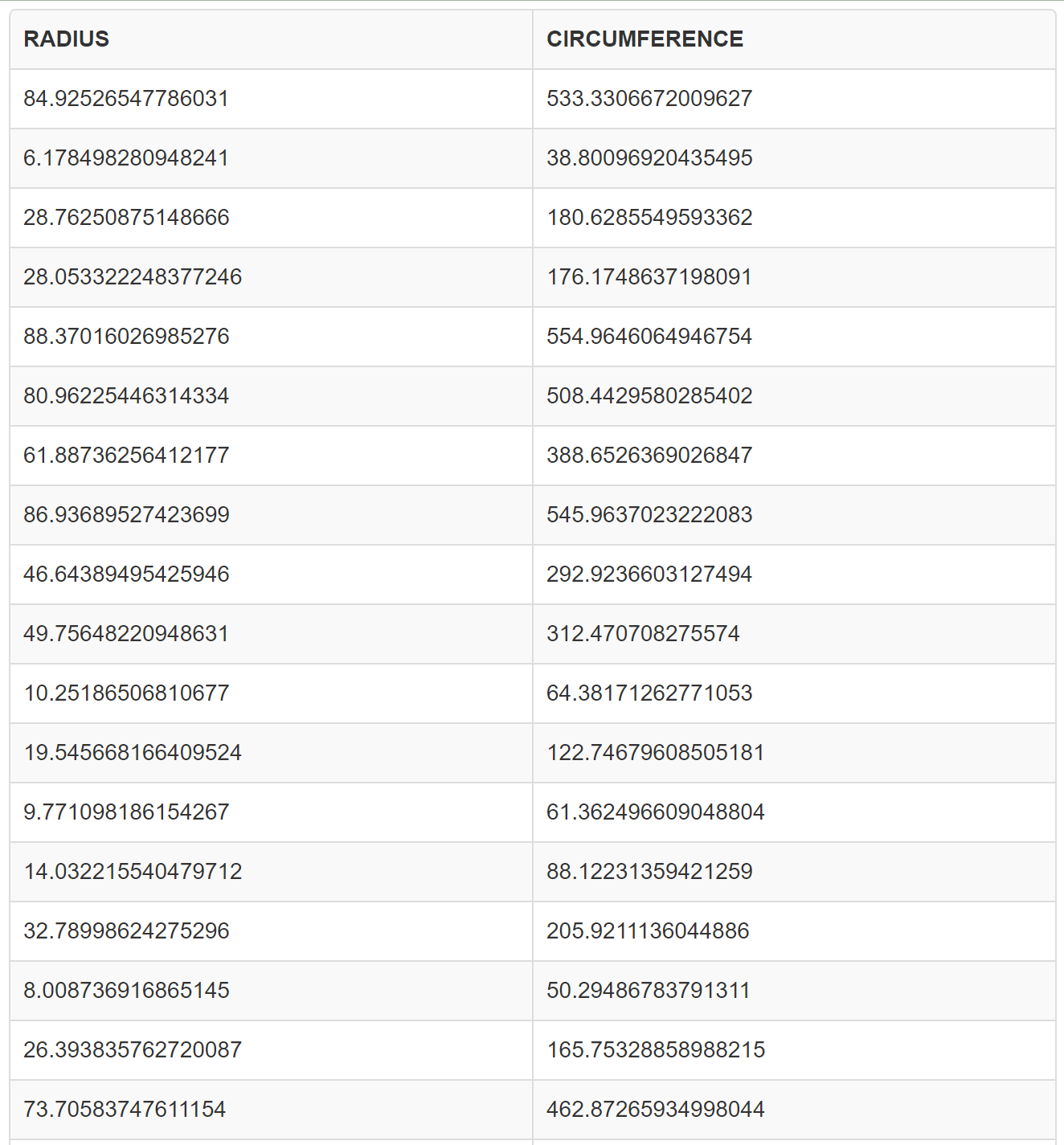
end;

/

select \* from circle

/

**Result:**



1. **Update the balance of each customer from a cust\_acct table showing withdrawal of Rs.1000/- as service charge provided that the customer balance shows at least Rs.1000/-.**

**Creation of table:**

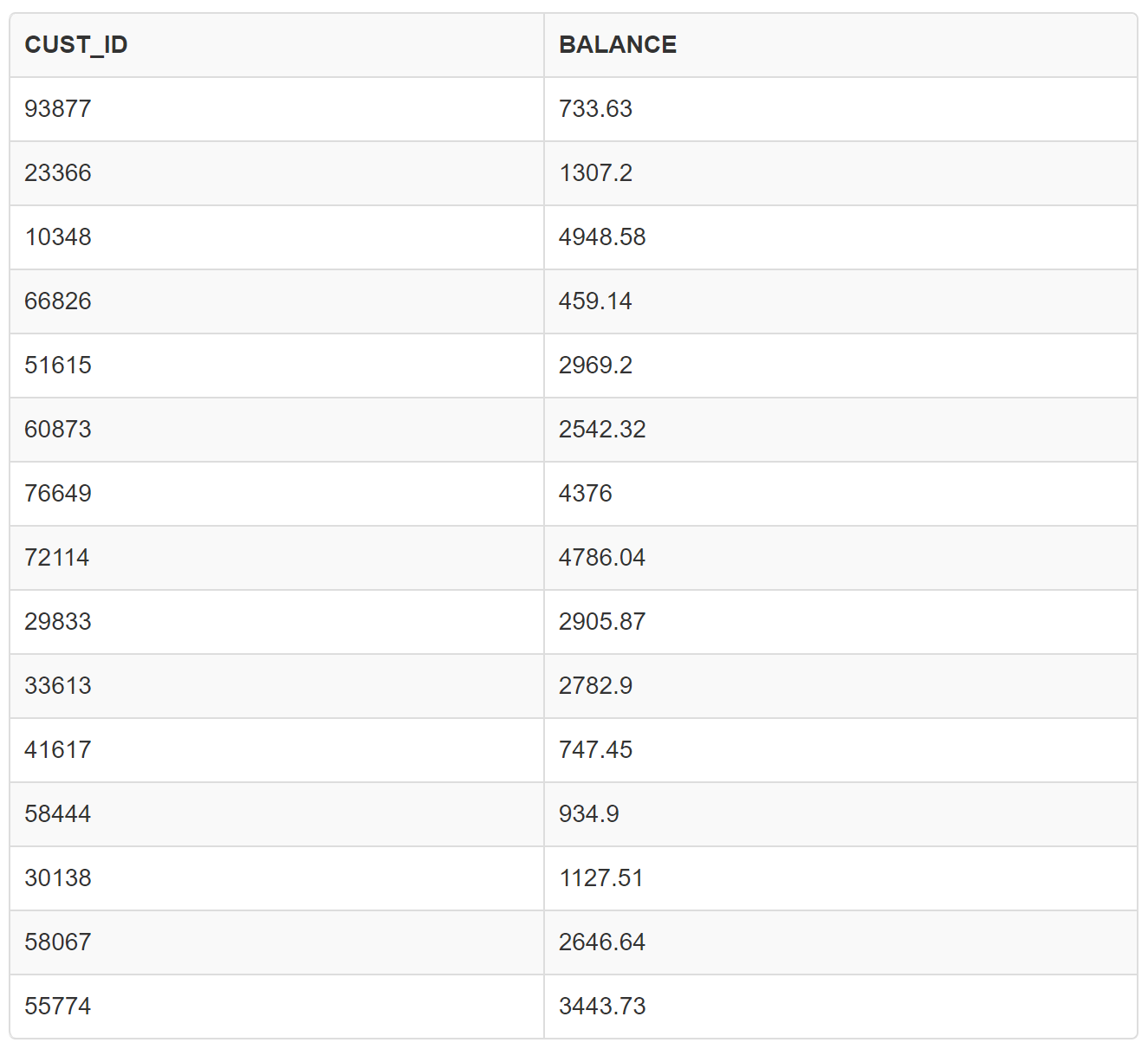
create table cust\_acct (

cust\_id number primary key,

balance number

);

**Snapshot of the table:**



**Pl/sql block for updation:**

declare

begin

update cust\_acct

set balance = balance - 1000

where balance >= 1000;

commit;

dbms\_output.put\_line('balances updated successfully.');

exception

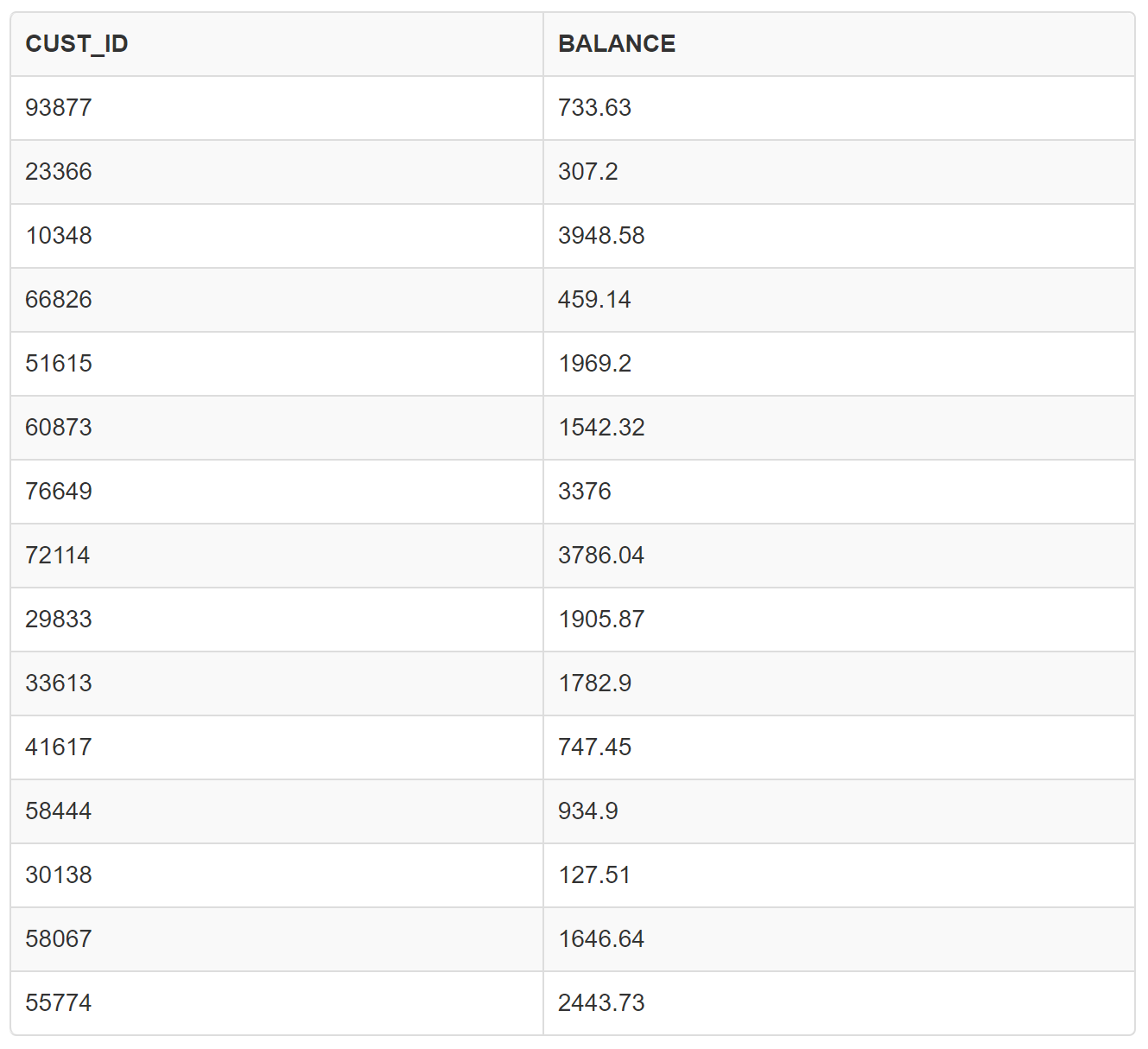
when others then

dbms\_output.put\_line('an error occurred while updating balances.');

end;

/

**Table after updation:**



1. **Update the salary of each employee from EMP table by 15% using cursor.**

**Creation of table EMP:**

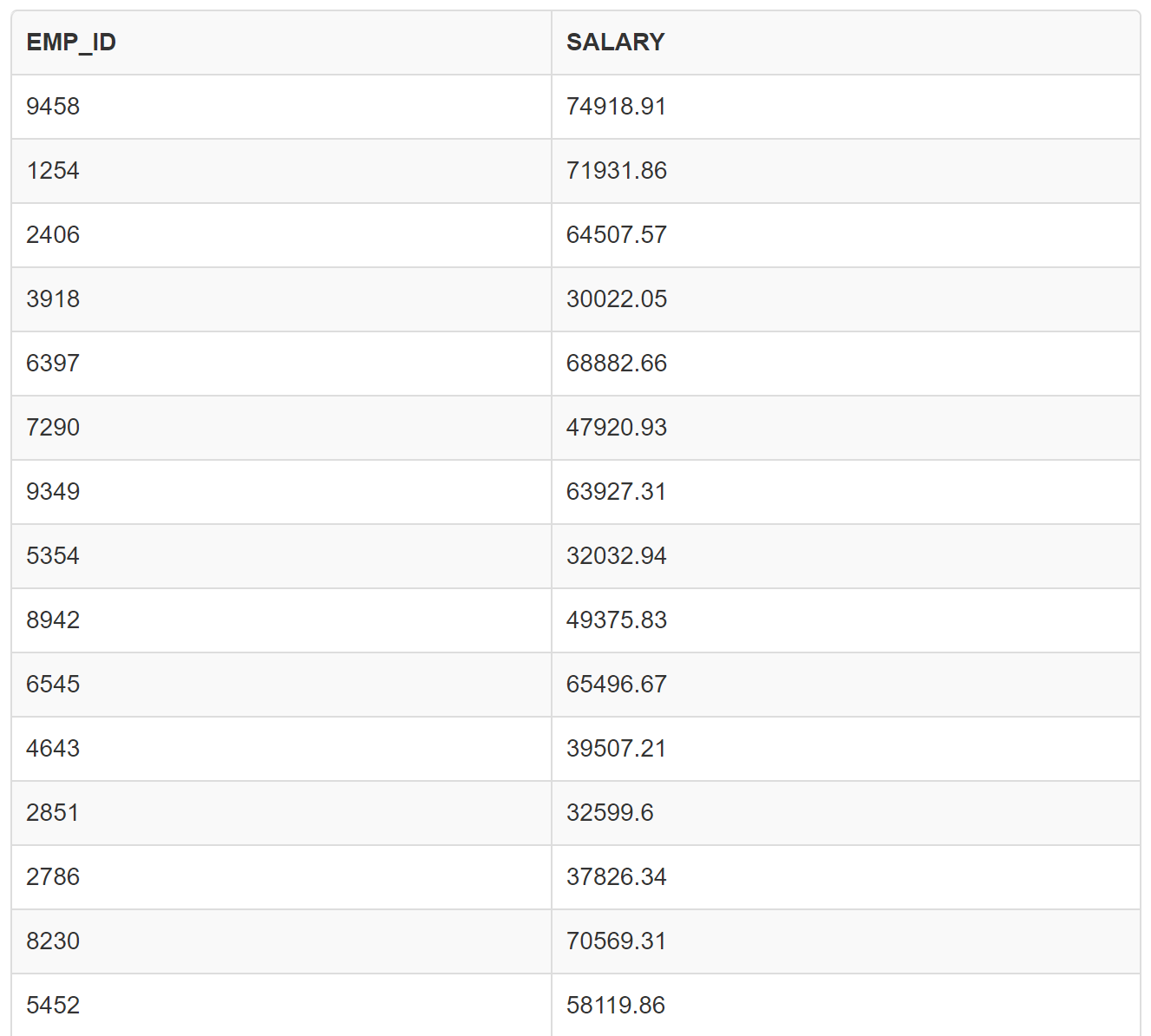
create table emp (

emp\_id number primary key,

salary number

);**­­­­**

**Snapshot of the table:**



**Pl/sql block for updation:**

declare

cursor emp\_cursor is

select emp\_id, salary

from emp;

v\_emp\_id emp.emp\_id%type;

v\_salary emp.salary%type;

begin

for emp\_rec in emp\_cursor loop

v\_emp\_id := emp\_rec.emp\_id;

v\_salary := emp\_rec.salary \* 1.15;

update emp

set salary = v\_salary

where emp\_id = v\_emp\_id;

end loop;

commit;

dbms\_output.put\_line('salaries updated successfully.');

exception

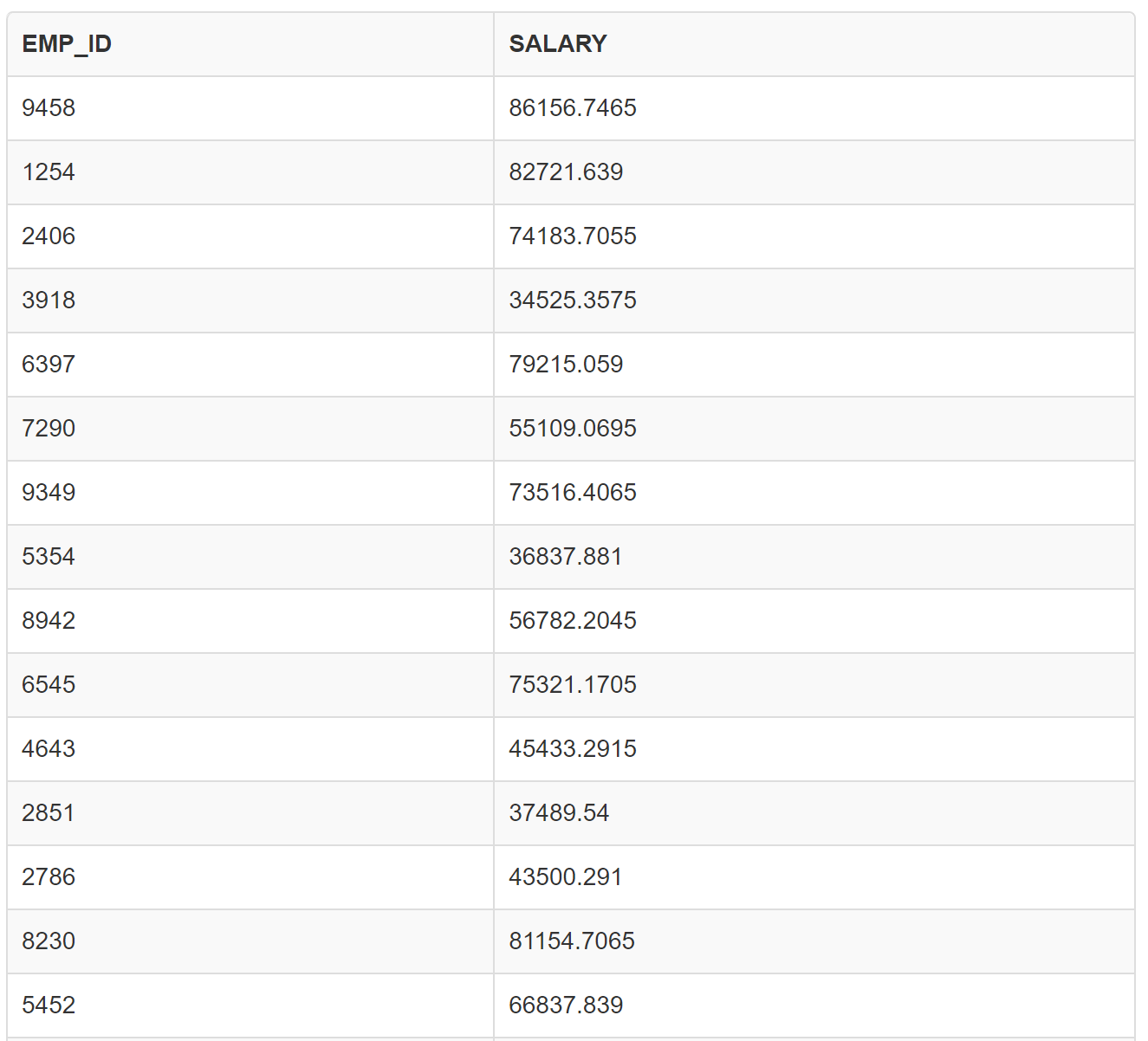
when others then

dbms\_output.put\_line('an error occurred while updating salaries.');

end;

/

**Table after updation:**



1. **Update the balance in the ITEM\_MSTR table each time a transaction takes place in the ITEM\_TR table. If this item\_id is already present in the ITEM\_MSTR table an update is performed to decrease the balance by the quantity specified in the ITEM\_TR table. If the item\_id is not present in the ITEM\_MSTR table, the tuple is to be inserted.**

**Creation of table ITEM\_MSTR:**

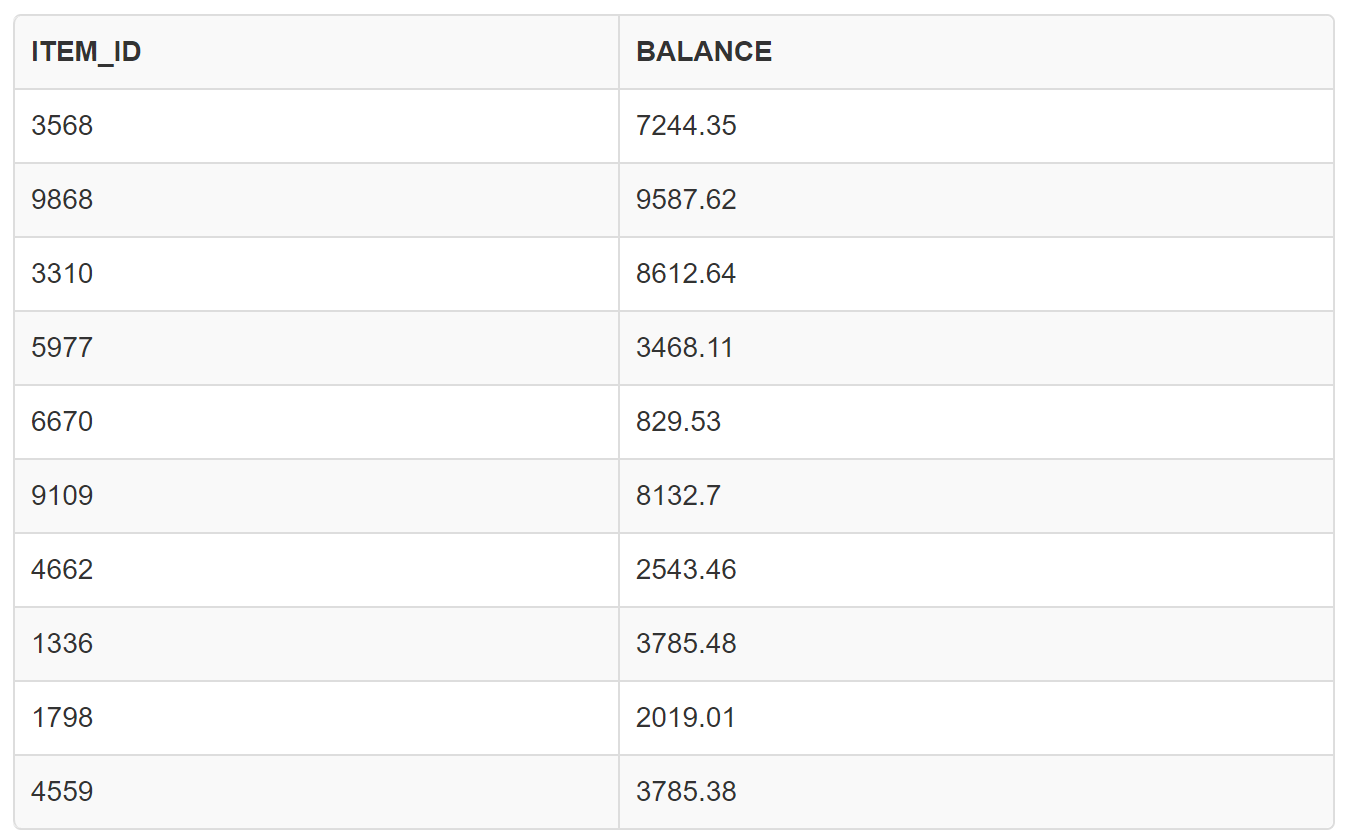
create table item\_mstr (

item\_id number primary key,

balance number

);

**Snapshot of table ITEM\_MSTR:**



**Creation of table ITEM\_TR:**

create table item\_tr (

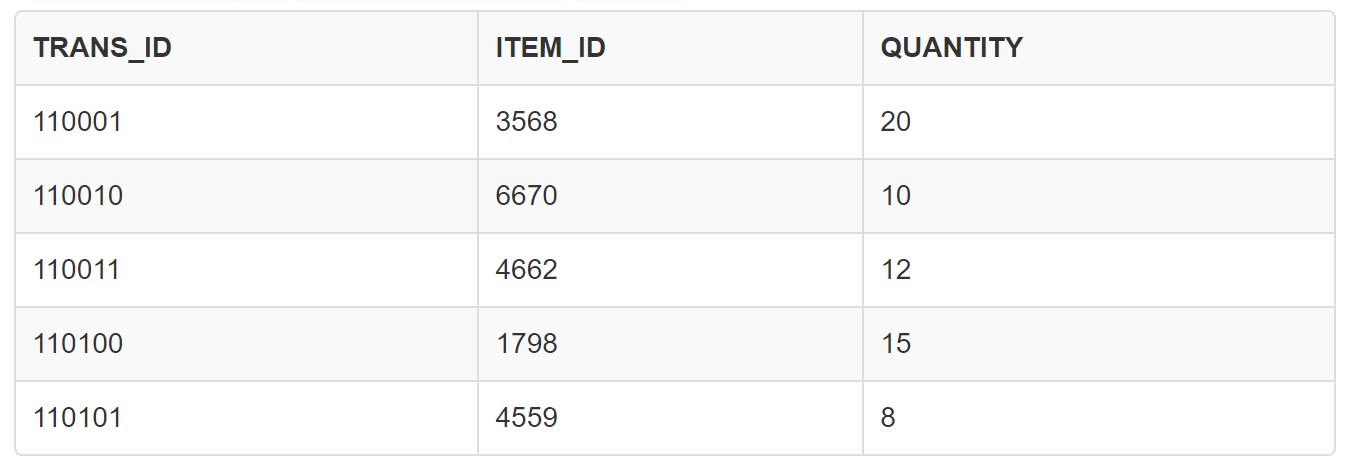
trans\_id number primary key,

item\_id number,

quantity number

);

**Snapshot of table ITEM\_TR:**



**Pl/sql block to update tables:**

declare

v\_item\_id number;

v\_quantity number;

v\_balance number;

begin

for trans\_rec in (select item\_id, quantity from item\_tr) loop

v\_item\_id := trans\_rec.item\_id;

v\_quantity := trans\_rec.quantity;

-- check if the item\_id exists in item\_mstr.

select balance into v\_balance

from item\_mstr

where item\_id = v\_item\_id;

if v\_balance is not null then

-- item exists, update the balance.

update item\_mstr

set balance = balance - v\_quantity

where item\_id = v\_item\_id;

else

-- item doesn't exist, insert a new record.

insert into item\_mstr (item\_id, balance) values (v\_item\_id, -v\_quantity);

end if;

end loop;

commit;

dbms\_output.put\_line('balances updated in item\_mstr table.');

exception

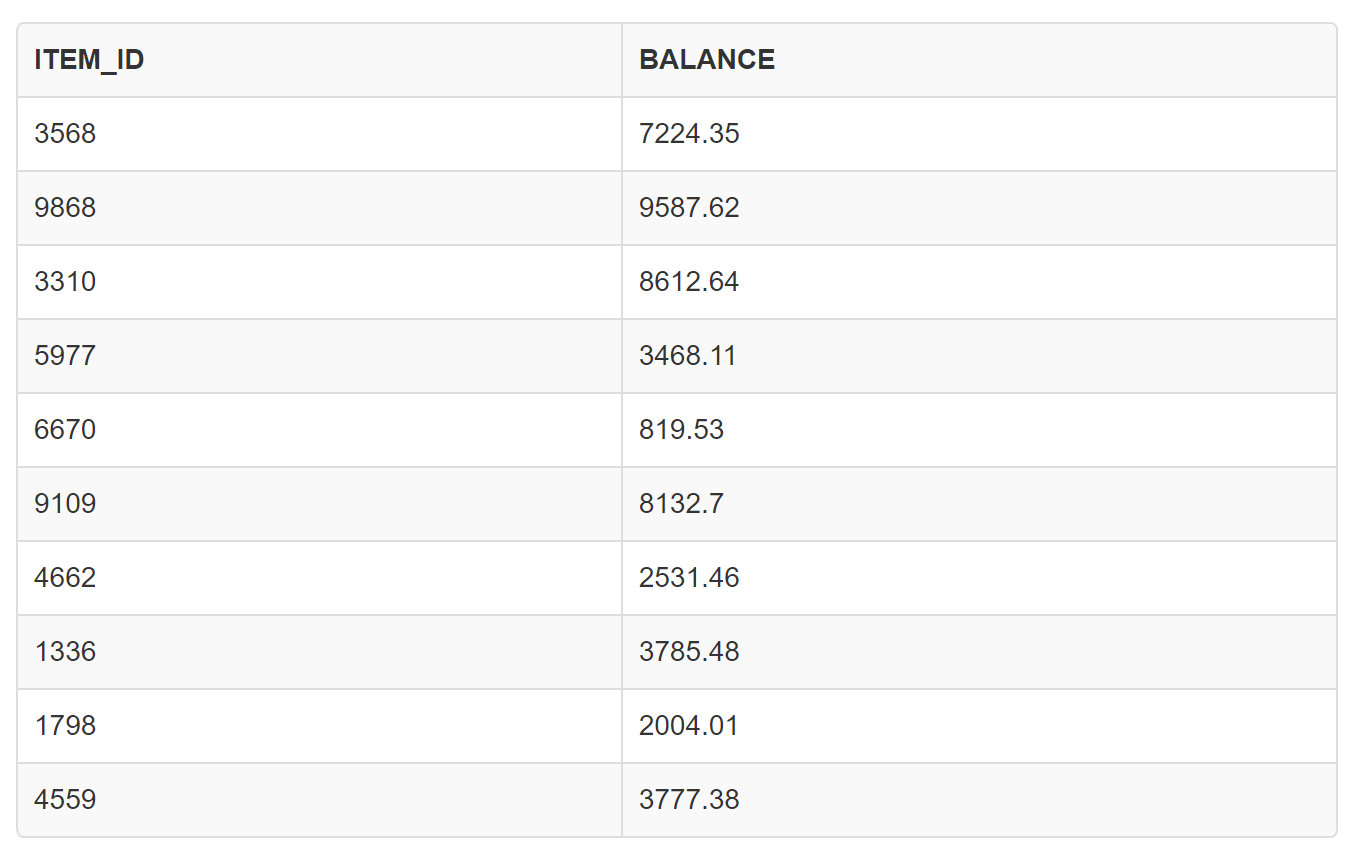
when others then

dbms\_output.put\_line('an error occurred while updating balances.');

end;

/

**Table after updation:**



1. **Write a PROCEDURE for raising salary of some employee by some amount. The PROCEDURE to be written may carry two parameters emp\_id and amt to be raised. Include two exceptions which will be raised when either emp\_id is not present or salary is NULL.**

**Creation of table:**

create table employees (

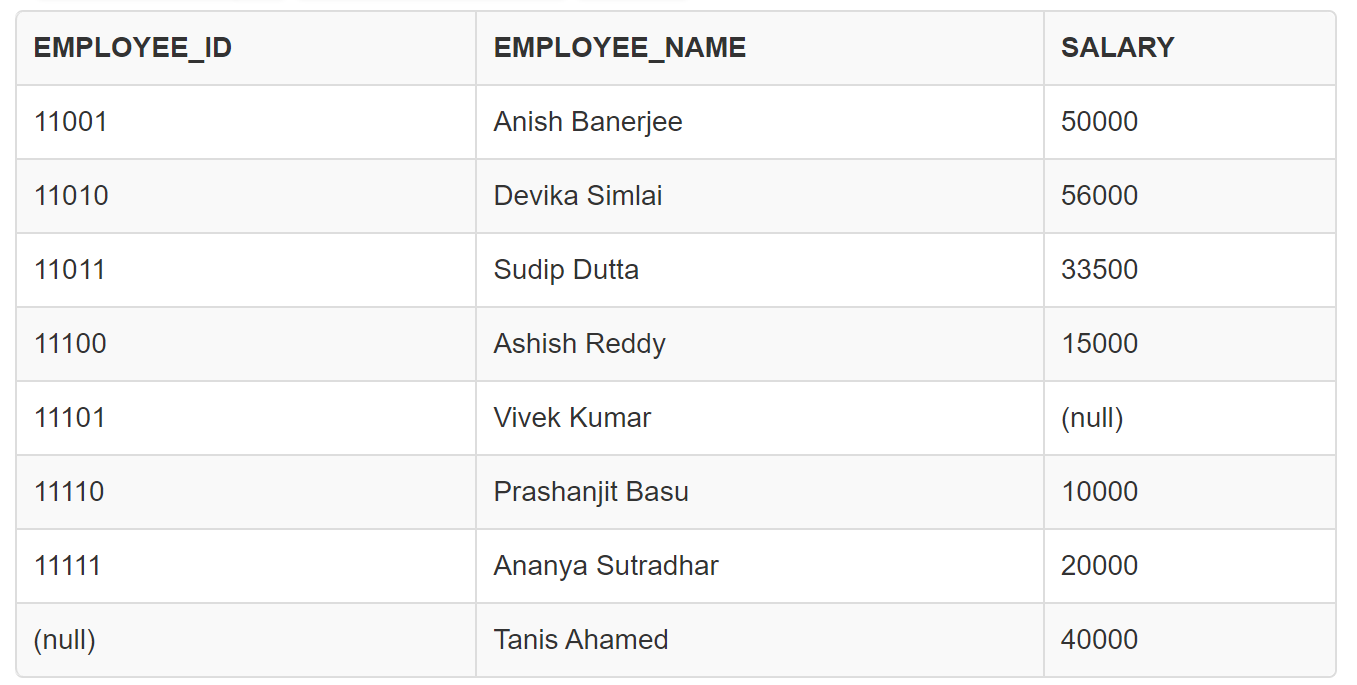
employee\_id number,

employee\_name varchar2(50),

salary number

);

**Snapshot of the table:**



**Procedure to raise salary:**

CREATE OR REPLACE PROCEDURE raise\_salary(p\_emp\_id IN NUMBER, p\_amt IN NUMBER) IS

v\_salary NUMBER;

BEGIN

SELECT salary INTO v\_salary

FROM employees

WHERE employee\_id = p\_emp\_id;

IF v\_salary IS NULL THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Employee ID not found.');

END IF;

IF v\_salary + p\_amt IS NULL THEN

RAISE\_APPLICATION\_ERROR(-20002, 'Cannot raise salary. Current salary is NULL.');

END IF;

UPDATE employees

SET salary = salary + p\_amt

WHERE employee\_id = p\_emp\_id;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Salary raised successfully.');

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Employee ID not found.');

WHEN OTHERS THEN

RAISE\_APPLICATION\_ERROR(-20000, 'An error occurred: ' || SQLCODE || ' - ' || SQLERRM);

END raise\_salary;

/

**Calling the procedure - I:**

begin

raise\_salary(11001, 5000);

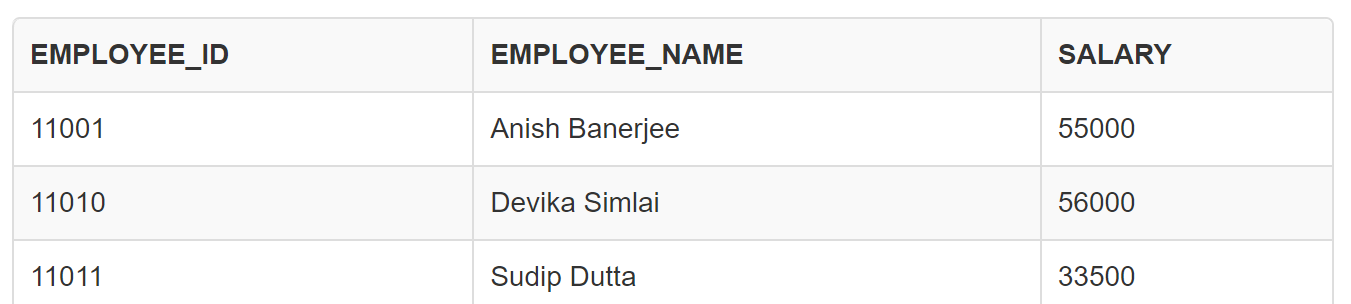
end;

/

select\* from employees

/

**Update in the table:**



**Calling the procedure - II:**

begin

raise\_salary(11010, 4000);

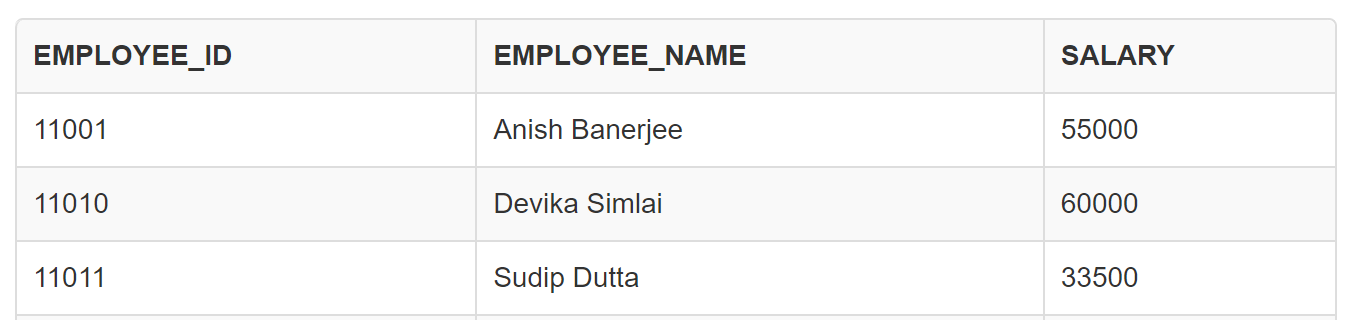
end;

/

select \* from employees

/

**Update in the table:**



**Calling the procedure where salary is null:**

begin

raise\_salary(11101, 5000);

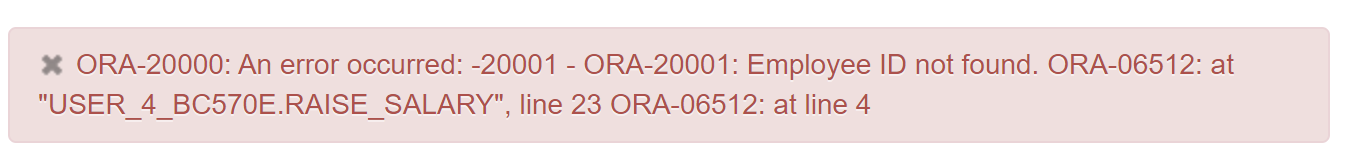
end;

/

select \* from employees

/

**Update in the table:**



**Calling the procedure where employee\_id is null:**

begin

raise\_salary(11000, 5000);

end;

/

select \* from employees

/

**Update in the table:**

