ACADEMIA DE STUDII ECONOMICE DIN BUCUREȘTI

Facultatea de Cibernetică, Statistică și Informatică Economică

**DBMS Project**

**Managing a chain of hotels**

Tudorie Marius Cosmin Teacher:

G series group 1065 Diaconiță Vlad

2020

**Contents:**

I:PL/SQL

a: Describing the problem and presenting the conceptual schema. 3

b: Interacting with Oracle server using SQL statements in PLSQL blocks; 4

c: Decision and loop control structures 6

d: Managing exceptions 7

e: Managing cursors 9

f: Functions, procedures, packages 12

g: Triggers 16

II: Apex

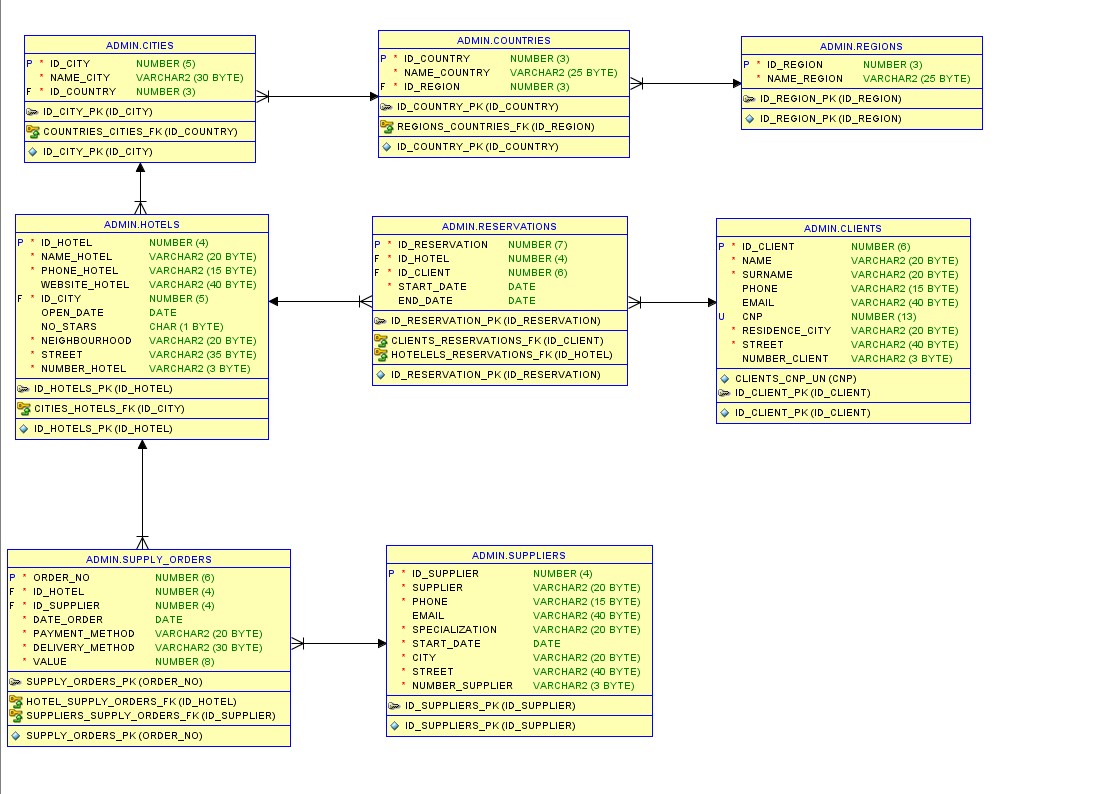
h: Forms 17

i: Reports 21

I: PL/SQL

a: Describing the problem and presenting the conceptual schema.

The project has the objective of keeping an organized evidence of an chain of hotels, bringing together elements like clients and suppliers dus allowing the user to quickly visualize key elements related to it’s travell business.

The database contain a total of eigth tables and has the following conceptual schema:

The eight tables are:

* Regions
* Countries
* Cities
* Hotels
* Clients
* Reservations
* Suppliers
* Supply Orders

b: Interacting with Oracle server using SQL statements in PLSQL blocks;

creating a copy of the table clients

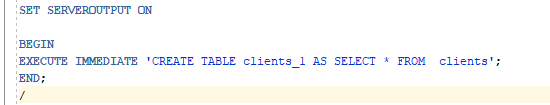
SET SERVEROUTPUT ON

BEGIN

EXECUTE IMMEDIATE 'CREATE TABLE clients\_1 AS SELECT \* FROM clients';

END;

/



Display the name, surname and phone number of the client with id 1 from the copy of the table we created

SET SERVEROUTPUT ON

DECLARE

v\_name clients\_1.name%type;

v\_surname clients\_1.surname%type;

v\_phone clients\_1.phone%type;

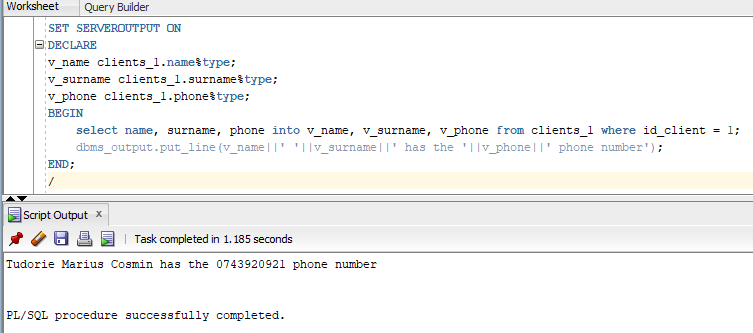
BEGIN

select name, surname, phone into v\_name, v\_surname, v\_phone from clients\_1 where id\_client = 1;

dbms\_output.put\_line(v\_name||' '||v\_surname||' has the '||v\_phone||' phone number');

END;

/



Add a new column Client\_type to the table clients\_1;

SET SERVEROUTPUT ON

DECLARE

v\_sql varchar(150);

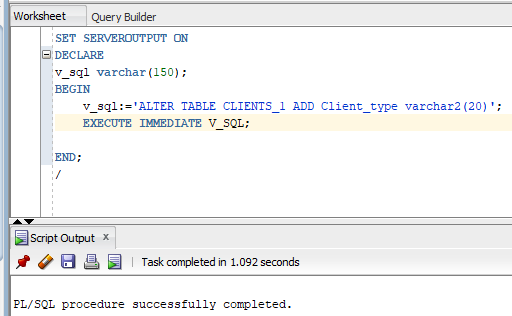
BEGIN

v\_sql:='ALTER TABLE CLIENTS\_1 ADD Client\_type varchar2(20)';

EXECUTE IMMEDIATE V\_SQL;

END;

/



Update client with id 1 to premium client type;

SET SERVEROUTPUT ON

DECLARE

v\_sql varchar(150);

v\_name clients\_1.name%type;

v\_type clients\_1.client\_type%type;

BEGIN

v\_sql:='UPDATE clients\_1 set client\_type = ''premium'' where id\_client=:1';

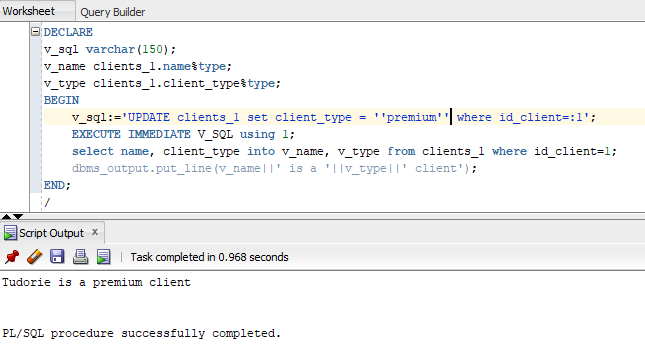
EXECUTE IMMEDIATE V\_SQL using 1;

select name, client\_type into v\_name, v\_type from clients\_1 where id\_client=1;

dbms\_output.put\_line(v\_name||' is a '||v\_type||' client');

END;

/



Drop the clients\_1 table;

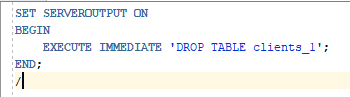
SET SERVEROUTPUT ON

BEGIN

EXECUTE IMMEDIATE 'DROP TABLE clients\_1';

END;

/



c: Decision and loop control structures

Diplay the name of the hotel on the supply order that has the value bigger than 100.000

SET SERVEROUTPUT ON

DECLARE

V\_HOTEL HOTELS.NAME\_HOTEL%TYPE;

V\_VALUE SUPPLY\_ORDERS.VALUE%TYPE;

i pls\_integer :=0;

BEGIN

for i in 1..10 LOOP

SELECT VALUE, NAME\_HOTEL INTO V\_VALUE, V\_HOTEL FROM SUPPLY\_ORDERS S JOIN HOTELS H ON S.ID\_HOTEL = H.ID\_HOTEL WHERE ORDER\_NO=i;

if V\_VALUE >100000 THEN

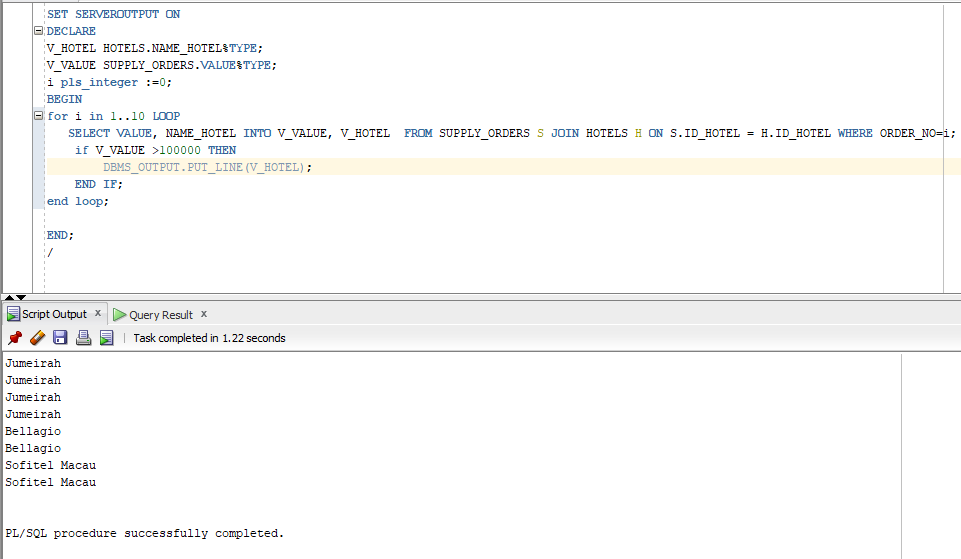
DBMS\_OUTPUT.PUT\_LINE(V\_HOTEL);

END IF;

end loop;

END;

/



Display the type of transport that suppliers have made

SET SERVEROUTPUT ON

DECLARE

V\_SUPPLIER SUPPLIERS.SUPPLIER%TYPE;

V\_TRANSPORT SUPPLY\_ORDERS.DELIVERY\_METHOD%TYPE;

i pls\_integer :=1;

BEGIN

LOOP

IF I>10 THEN

EXIT;

END IF;

SELECT DELIVERY\_METHOD, SUPPLIER INTO V\_TRANSPORT, V\_SUPPLIER FROM SUPPLY\_ORDERS O JOIN SUPPLIERS S ON O.ID\_SUPPLIER = S.ID\_SUPPLIER WHERE ORDER\_NO=i;

CASE V\_TRANSPORT

WHEN 'Airline' then dbms\_output.put\_line('airline delivery -> '|| V\_SUPPLIER );

WHEN 'Shipping' THEN dbms\_output.put\_line('shipping -> '|| V\_SUPPLIER );

WHEN 'Rail' THEN dbms\_output.put\_line('Rail ->'|| V\_SUPPLIER );

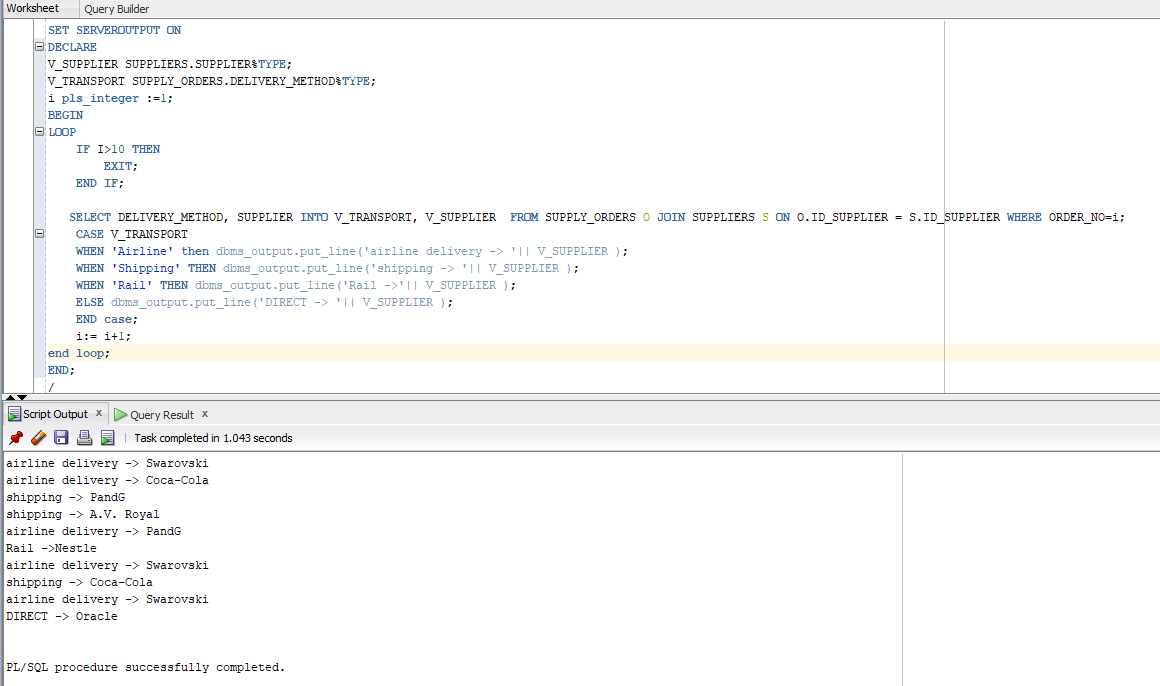
ELSE dbms\_output.put\_line('DIRECT -> '|| V\_SUPPLIER );

END case;

i:= i+1;

end loop;

END;



D: Managing exceptions

Display hotel information of the row with id read from the keybord managing no found exceptions

SET SERVEROUTPUT ON

ACCEPT insert\_no PROMPT 'INSERT HOTEL ID:';

DECLARE

v\_id number(3):=&insert\_no;

v\_name hotels.name\_hotel%type;

v\_phone hotels.phone\_hotel%type;

v\_stars hotels.no\_stars%type;

begin

select name\_hotel, phone\_hotel, no\_stars into v\_name, v\_phone, v\_stars from hotels where id\_hotel = v\_id;

dbms\_output.put\_line('hotel '||v\_name||' has '||v\_stars||' stars. Contact at:'||v\_phone);

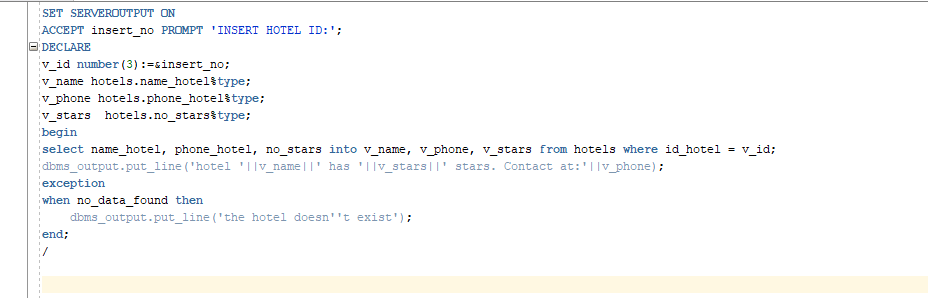
exception

when no\_data\_found then

dbms\_output.put\_line('the hotel doesn''t exist');

end;

/



Trying to display the name of the hotel situated on Zayed Road street. In case of multiple hotels the exception will be treated;

SET SERVEROUTPUT ON

DECLARE

V\_HOTEL HOTELS.NAME\_HOTEL%TYPE;

BEGIN

SELECT NAME\_HOTEL INTO V\_HOTEL FROM HOTELS WHERE STREET='Zayed Road';

dbms\_output.put\_line('the hotel from the zayed street is '||v\_hotel);

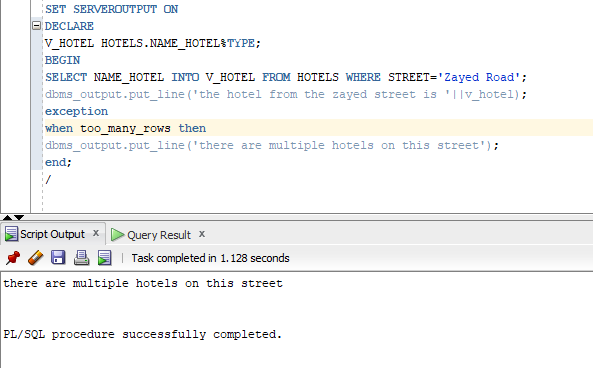
exception

when too\_many\_rows then

dbms\_output.put\_line('there are multiple hotels on this street');

end;

/



Raise price with 20% to all the deliveries if the delivery method is by Road raise custom exception if no rows affected;

SET SERVEROUTPUT ON

DECLARE

EX EXCEPTION;

BEGIN

UPDATE SUPPLY\_ORDERS SET VALUE= VALUE\*1.2 WHERE DELIVERY\_METHOD = 'Road';

if sql%rowcount=0 then

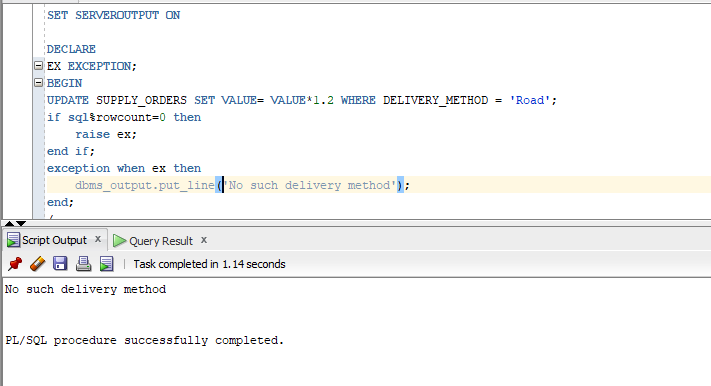
raise ex;

end if;

exception when ex then

dbms\_output.put\_line('No such delivery method');

end;



e: Managing cursors

Parameter cursor looping trough countries for each region:

SET SERVEROUTPUT ON

DECLARE

CURSOR R IS SELECT \* FROM REGIONS ORDER BY 1;

CURSOR C(P\_ID\_REGION NUMBER) IS SELECT NAME\_COUNTRY FROM COUNTRIES WHERE ID\_REGION = P\_ID\_REGION;

t c%rowtype;

BEGIN

FOR I IN R LOOP

OPEN C(I.ID\_REGION);

FETCH C INTO T;

IF C%FOUND THEN

CLOSE C;

DBMS\_OUTPUT.PUT\_LINE('Region '|| I.NAME\_REGION||' has id '||I.ID\_REGION);

FOR J IN C(I.ID\_REGION) LOOP

DBMS\_OUTPUT.PUT\_LINE(' '||J.NAME\_COUNTRY);

END LOOP;

ELSE

CLOSE C;

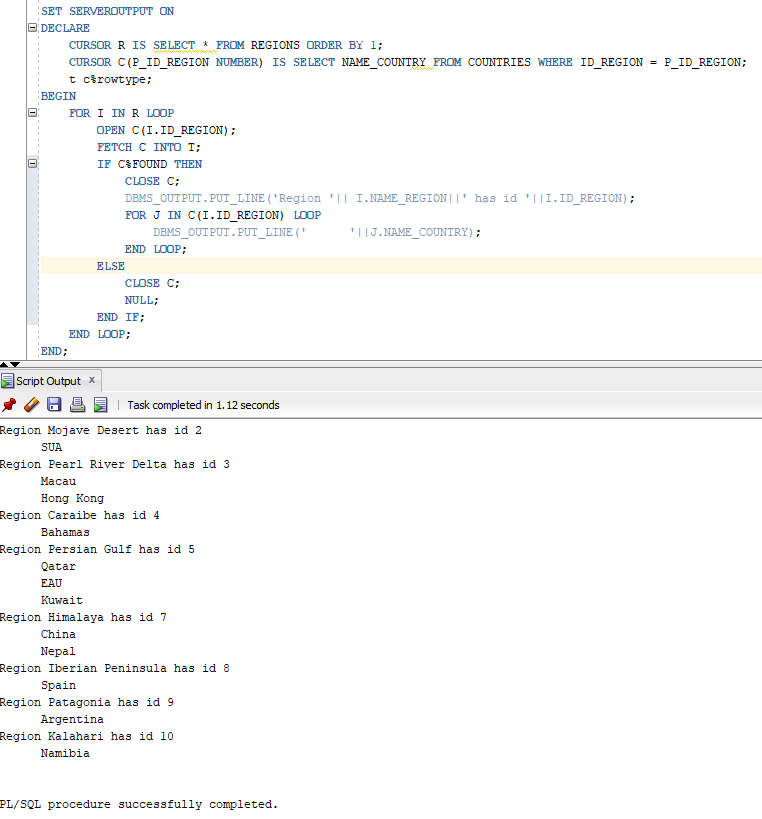
NULL;

END IF;

END LOOP;

END;

/



Set value of website for keybord selected id value

SET SERVEROUTPUT ON

BEGIN

UPDATE HOTELS SET WEBSITE\_HOTEL ='WWW.HOTEL.COM' WHERE ID\_HOTEL = &ID;

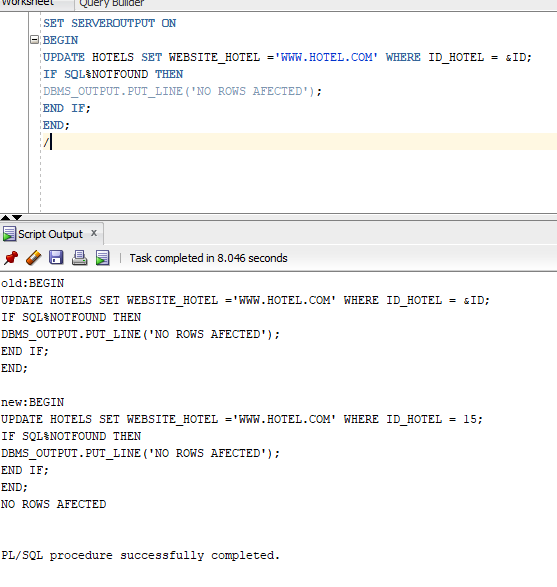
IF SQL%NOTFOUND THEN

DBMS\_OUTPUT.PUT\_LINE('NO ROWS AFECTED');

END IF;

END;

/



Using a cursor that gets a parameter the type of payment, all the orders with that method of payment will be displayed

SET SERVEROUTPUT ON

DECLARE

CURSOR C(P\_PAYMENT VARCHAR2) IS SELECT \* FROM SUPPLY\_ORDERS WHERE PAYMENT\_METHOD = P\_PAYMENT;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('CASH PAYED ORDERS');

FOR I IN C('Cash') LOOP

DBMS\_OUTPUT.PUT\_LINE('ORDER WITH ID:'|| I.ORDER\_NO||' FROM THE DATE OF ' ||I.DATE\_ORDER ||' HAS BEEN PAYED '||I.PAYMENT\_METHOD);

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('BANK TRANSFER PAYED ORDERS');

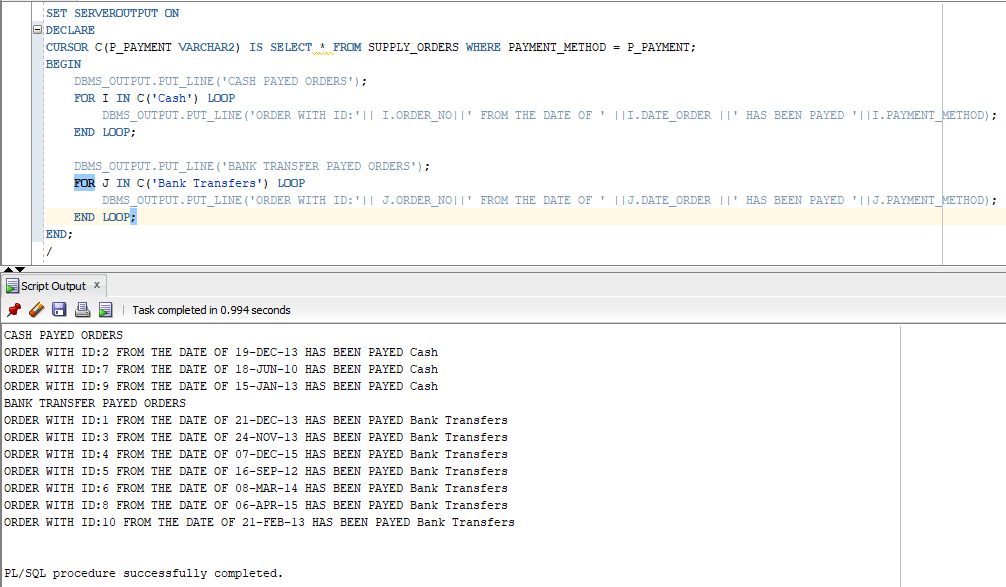
FOR J IN C('Bank Transfers') LOOP

DBMS\_OUTPUT.PUT\_LINE('ORDER WITH ID:'|| J.ORDER\_NO||' FROM THE DATE OF ' ||J.DATE\_ORDER ||' HAS BEEN PAYED '||J.PAYMENT\_METHOD);

END LOOP;

END;

/



f: Functions, procedures, packages

Function that returns the number of stars of a hotel given it’s name;

CREATE OR REPLACE FUNCTION get\_stars(p\_hotel HOTELS.NAME\_HOTEL%TYPE)

RETURN CHAR

IS

V\_NO\_STARS HOTELS.NO\_STARS%TYPE;

BEGIN

SELECT NO\_STARS INTO V\_NO\_STARS FROM HOTELS WHERE NAME\_HOTEL=P\_HOTEL;

RETURN V\_NO\_STARS;

END;

/

DECLARE

V\_STARS HOTELS.NO\_STARS%TYPE;

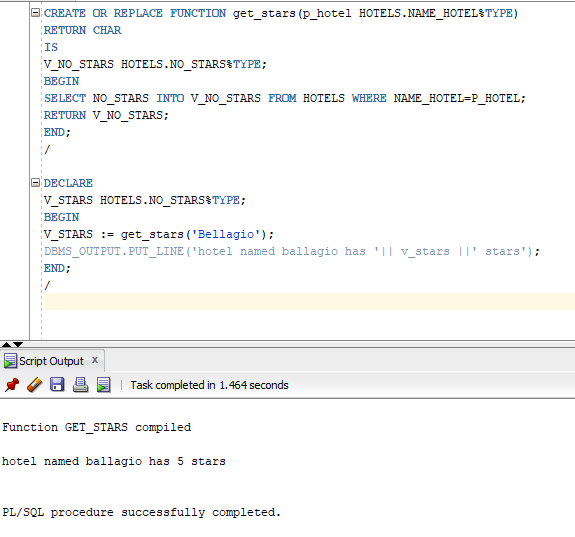
BEGIN

V\_STARS := get\_stars('Bellagio');

DBMS\_OUTPUT.PUT\_LINE('hotel named ballagio has '|| v\_stars ||' stars');

END;

/



Procedure that sets the website of a given gotel to a given website:

CREATE OR REPLACE PROCEDURE set\_website

(p\_hotel IN HOTELS.NAME\_HOTEL%TYPE, p\_website IN VARCHAR2)

IS

v\_website HOTELS.WEBSITE\_HOTEL%TYPE;

BEGIN

UPDATE HOTELS

SET WEBSITE\_HOTEL = p\_website

WHERE NAME\_HOTEL = p\_hotel;

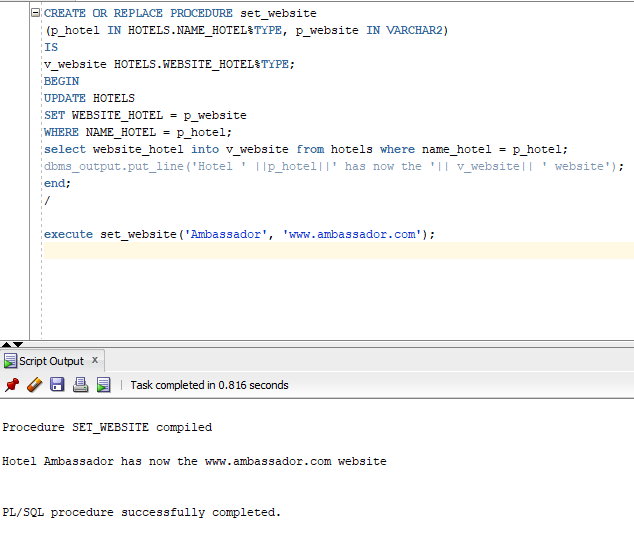
select website\_hotel into v\_website from hotels where name\_hotel = p\_hotel;

dbms\_output.put\_line('Hotel ' ||p\_hotel||' has now the '|| v\_website|| ' website');

end;

/

execute set\_website('Ambassador', 'www.ambassador.com');



Create a package that contains a procedure that computes tha average value of an order and a function that returns the order date of a given id;

CREATE OR REPLACE PACKAGE orders\_pack

as

procedure average\_value (avg\_val out number);

function get\_date (p\_id supply\_orders.order\_no%type) return date;

end;/

CREATE OR REPLACE PACKAGE body orders\_pack

as

procedure average\_value (avg\_val out number)

as

begin

select avg(value)into avg\_val from supply\_orders;

end;

function get\_date (p\_id supply\_orders.order\_no%type) return date

as

v\_date date;

begin

select Date\_order into v\_date from supply\_orders where order\_no = p\_id;

return v\_date;

exception

when no\_data\_found then DBMS\_OUTPUT.PUT\_LINE ('no order with that id'); return current\_date;

end;

end;

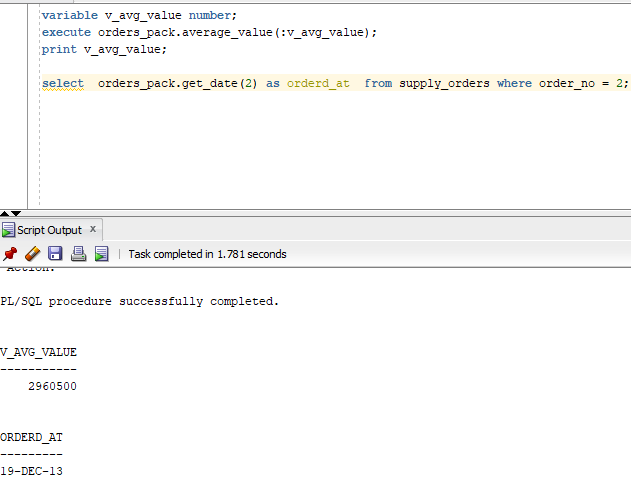
/

variable v\_avg\_value number;

execute orders\_pack.average\_value(:v\_avg\_value);

print v\_avg\_value;

select orders\_pack.get\_date(2) as orderd\_at from supply\_orders where order\_no = 2;



g: Triggers

a trigger that checks that the start date is always smaller than the end date in the reservations table;

create or replace trigger check\_date

before update on reservations

for each row

begin

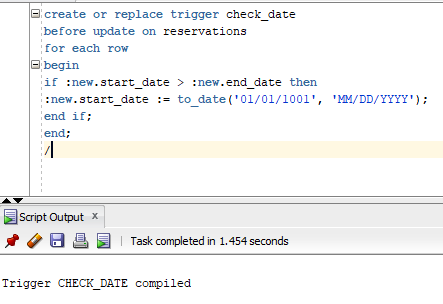
if :new.start\_date > :new.end\_date then

:new.start\_date := to\_date('01/01/1001', 'MM/DD/YYYY');

end if;

end;

/



II: APEX

Link Application: <https://tcufrwozwuuudbb-sgbdproject.adb.uk-london-1.oraclecloudapps.com/ords/f?p=102:31:5480754359418:::::>

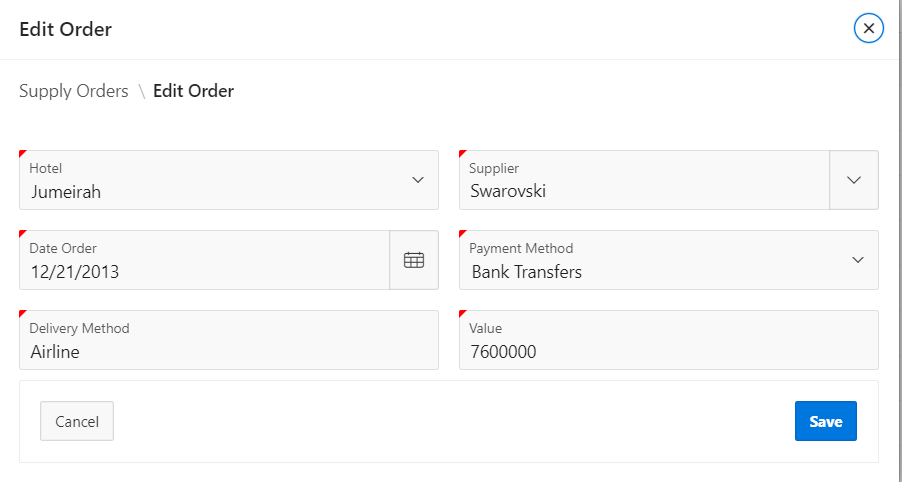
Username: TUDORIE MARIUS

Password: Tudoriemarius1

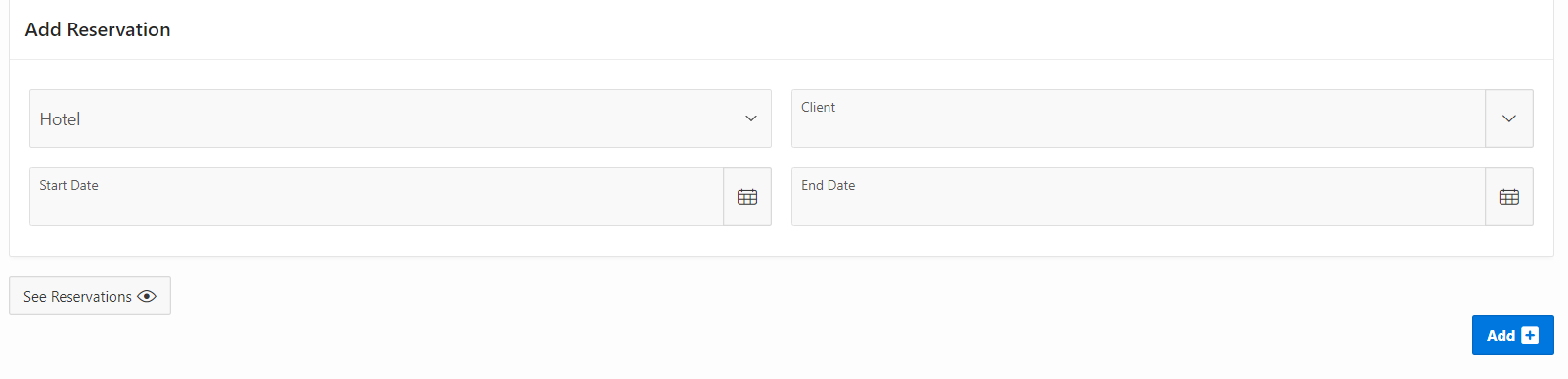
Workspace: TUDORIE MARIUS

H: Forms

Form that edits and adds new order:

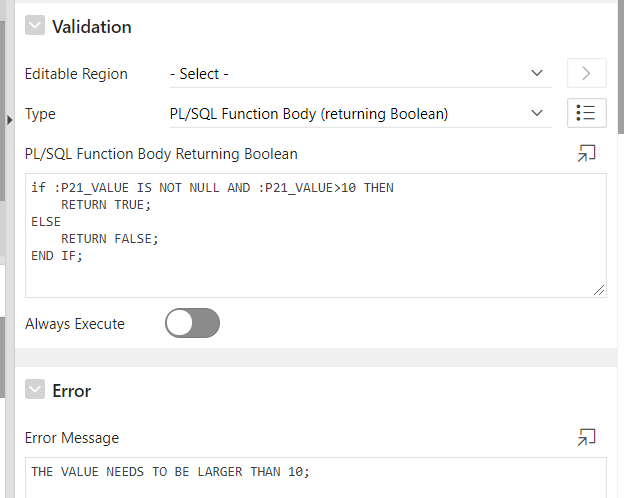


Form that adds new reservation:



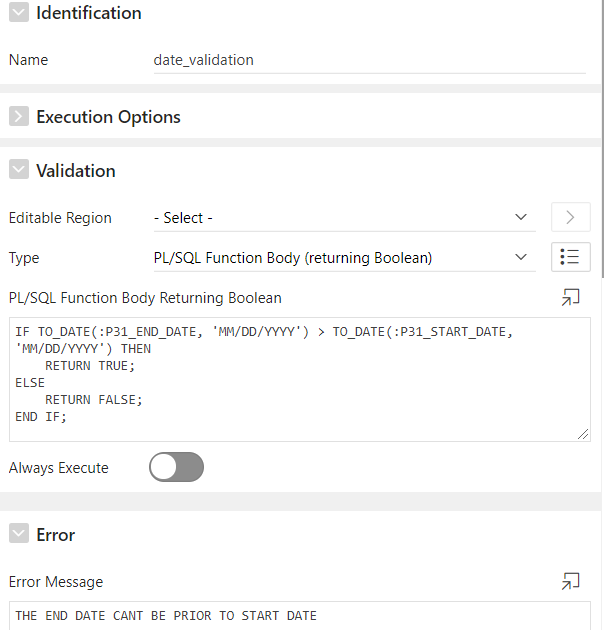
Validations

Validation on value field of the edit/add order form, checks if the value is not null and larger than 0.



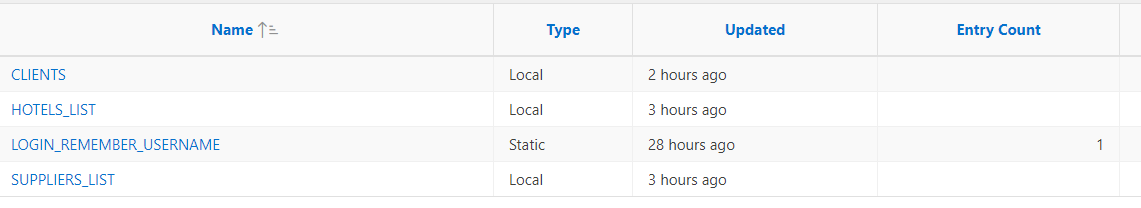
All the other fiels of this form have also not null validation.

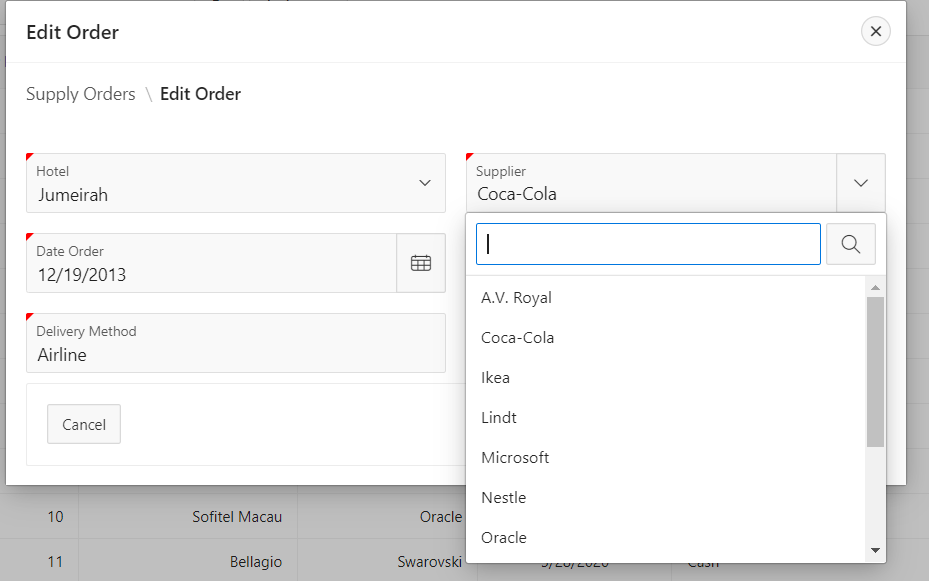
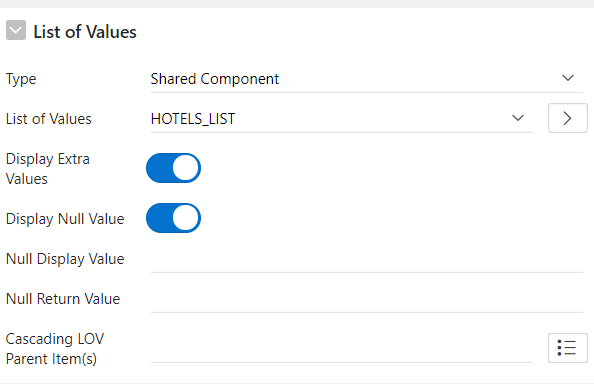
Validation maked on the end date field of the add reservation form checks weather the end date is smaller the start date.

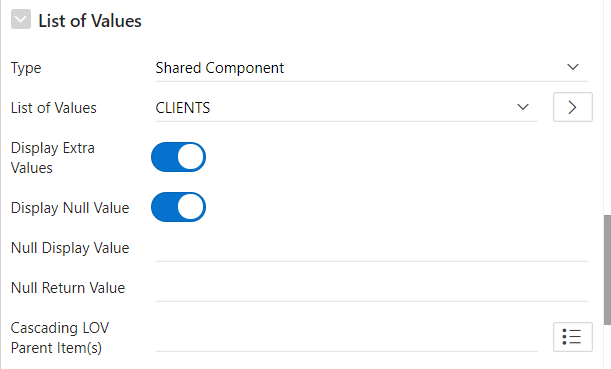
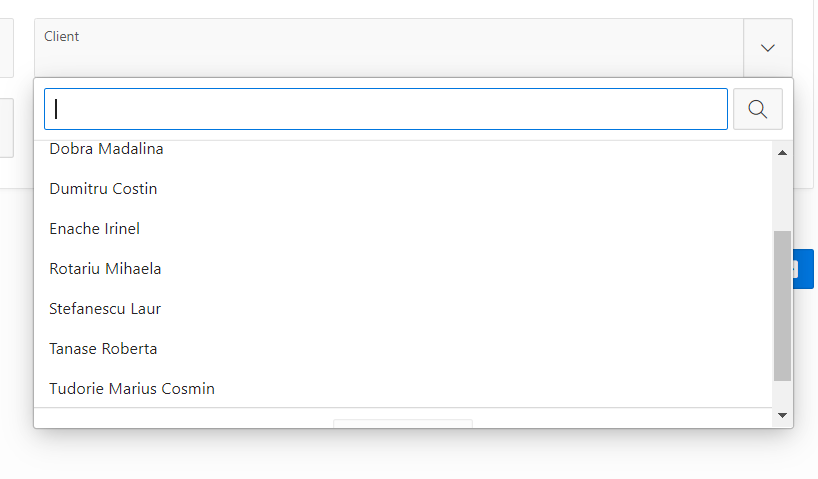
 All the other fiels have not null validation

Lov

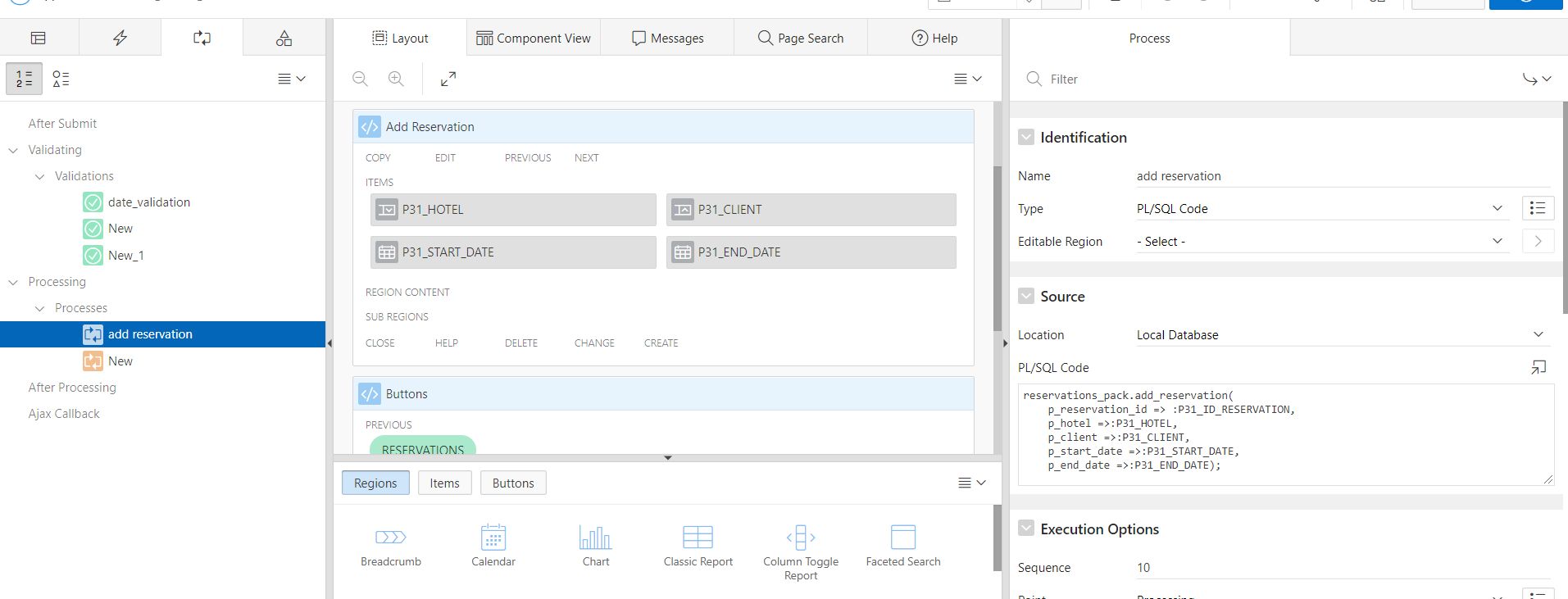
Shared components lists

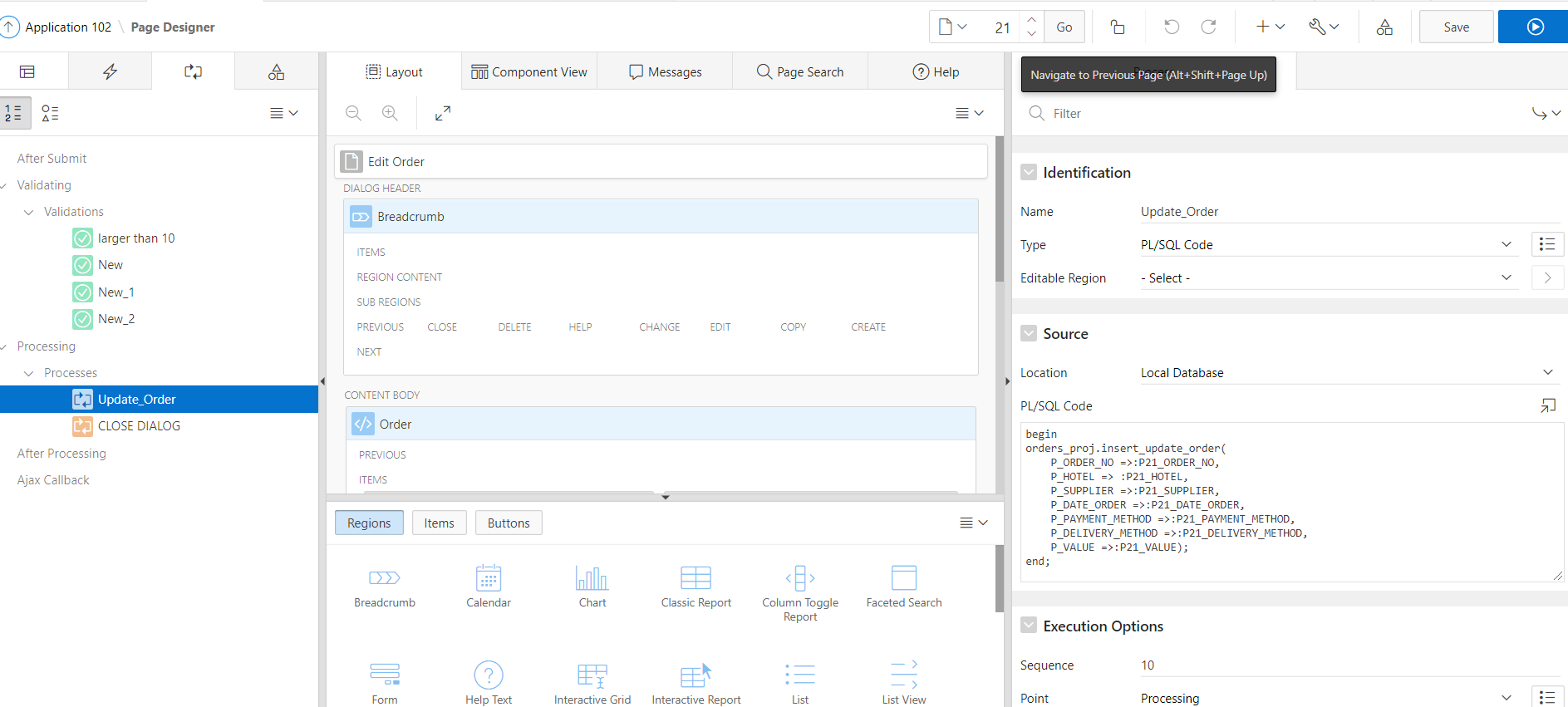




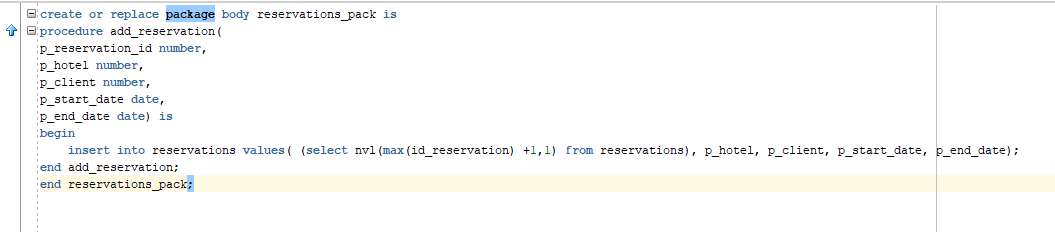


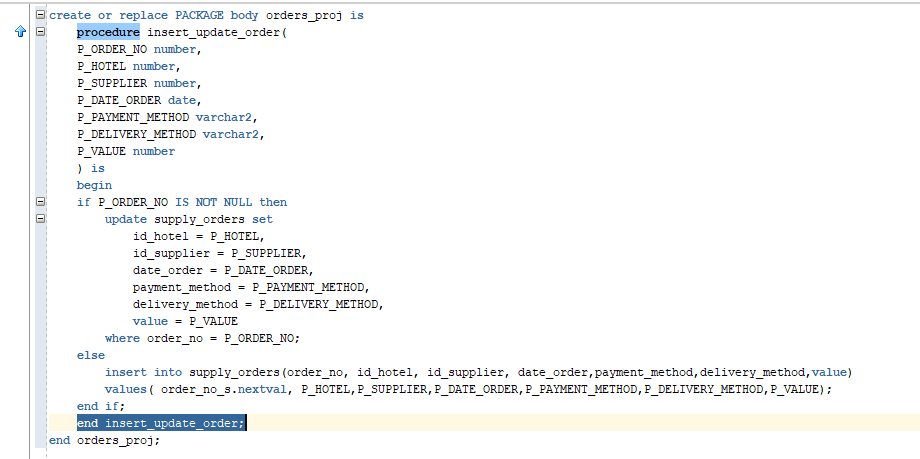
Dynamic actions/ processes





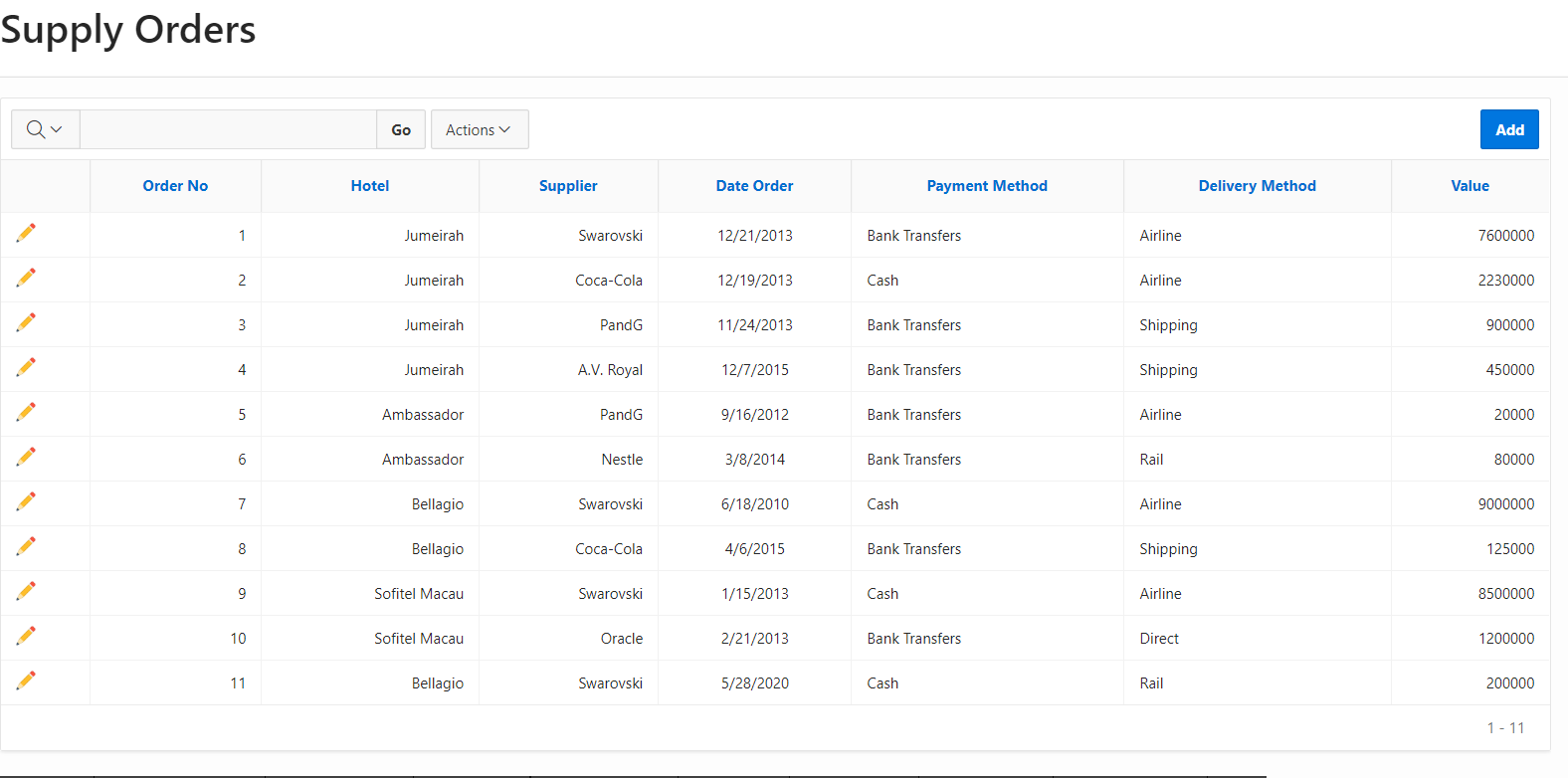
Packages for processes:





Reports:

Interactive:



Classic:

