**Zhumagali Kanat IT2-2003**

**Lab5. Creating Functions and Packages**

1. Create at least 3 functions for your database. You may use the anonymous blocks you created in Lab 2 as a basis.
2. At least 2 functions must be with parameters.
3. Combine your procedures (Lab 4) and functions (Lab 5) into at least 2 packages. Procedures and functions of one package must be logically related.
4. During the defense you must demonstrate the execution of procedures and functions.

create or replace function get\_movies return varchar2(4000)

is

movie\_info VARCHAR2(4000);

begin

for sal\_record in (select movie\_id, film\_name, duration, year\_of\_release from movie where YEAR\_OF\_RELEASE > 2000 order by year\_of\_release ASC) LOOP

movie\_info := movie\_info || 'Movie id : ' || sal\_record.movie\_id || ' Film name : ' || sal\_record.film\_name || ' Duration: ' || sal\_record.duration || ' Year of release: ' || sal\_record.year\_of\_release || CHR(10);

end loop;

return movie\_info;

end;

declare

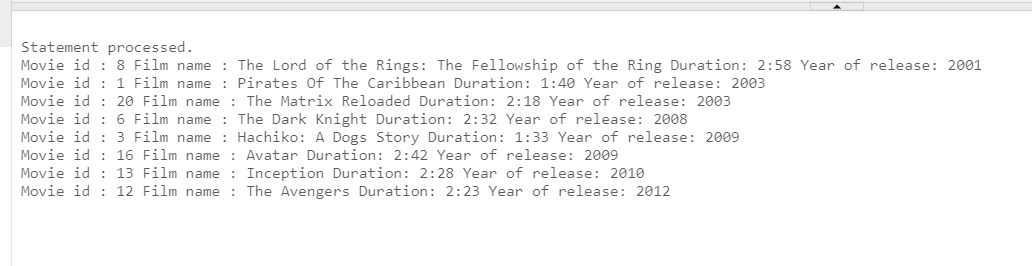
movie\_list VARCHAR2(4000);

begin

movie\_list := get\_movies();

dbms\_output.put\_line(movie\_list);

end;



create or replace function get\_movie\_info (

m\_id movie.movie\_id%TYPE

)

return varchar2

is

m\_name movie.film\_name%TYPE;

m\_duration movie.duration%TYPE;

m\_year movie.year\_of\_release%TYPE;

movie\_info VARCHAR2(4000); -- Adjust the size based on your actual requirements

begin

select film\_name, duration, year\_of\_release into m\_name, m\_duration, m\_year from movie where movie\_id = m\_id;

movie\_info := 'Movie name: ' || m\_name || ' duration: ' || m\_duration || ' year of release: ' || m\_year;

return movie\_info;

end get\_movie\_info;

declare

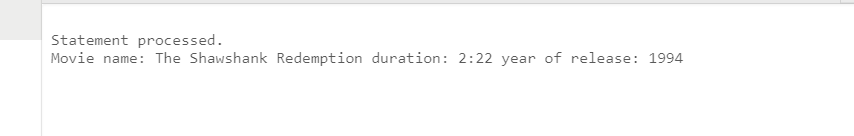
m\_info VARCHAR2(4000);

begin

m\_info := get\_movie\_info(4);

dbms\_output.put\_line(m\_info);

end;



create or replace function get\_genre\_movies(my\_genre genre.genre\_name%TYPE)

return varchar2

is

genre\_info VARCHAR2(4000);

begin

for genre\_rec in (select genre\_id, genre\_name from genre where genre\_name = my\_genre) loop

genre\_info := genre\_info || genre\_rec.genre\_id || ' Genre name: ' || genre\_rec.genre\_name || CHR(10);

for movie\_genre\_info\_rec in (select m.movie\_id, m.film\_name, m.year\_of\_release from movie m join movie\_genre mg ON mg.movie\_id = m.movie\_id where mg.genre\_id = genre\_rec.genre\_id) loop

genre\_info := genre\_info ||

'Movie id: ' || movie\_genre\_info\_rec.movie\_id ||

' Film name: ' || movie\_genre\_info\_rec.film\_name ||

' Year of release: ' || movie\_genre\_info\_rec.year\_of\_release || CHR(10);

end loop;

end loop;

return genre\_info;

end get\_genre\_movies;

declare

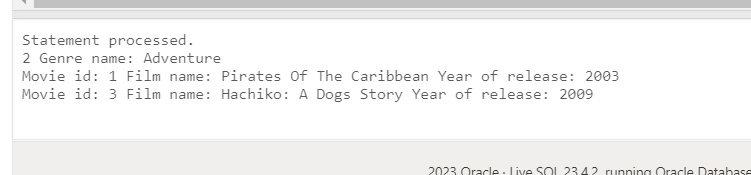
genre\_info\_result VARCHAR2(4000);

begin

genre\_info\_result := get\_genre\_movies('Adventure');

dbms\_output.put\_line(genre\_info\_result);

end;



create or replace package mov\_pkg is

procedure print\_movies;

function get\_movies return varchar2;

end mov\_pkg;

create or replace package body mov\_pkg is

procedure print\_movies is

begin

for sal\_record in (select movie\_id, film\_name, duration, year\_of\_release from movie where YEAR\_OF\_RELEASE > 2000 order by year\_of\_release asc) loop

dbms\_output.put\_line('Movie id : ' || sal\_record.movie\_id || ' Film name : ' || sal\_record.film\_name || ' Duration: ' || sal\_record.duration || ' Year of release: ' || sal\_record.year\_of\_release);

end loop;

end;

function get\_movies return varchar2

is

movie\_info VARCHAR2(4000);

begin

for sal\_record in (select movie\_id, film\_name, duration, year\_of\_release from movie where YEAR\_OF\_RELEASE > 2000 order by year\_of\_release ASC) LOOP

movie\_info := movie\_info || 'Movie id : ' || sal\_record.movie\_id || ' Film name : ' || sal\_record.film\_name || ' Duration: ' || sal\_record.duration || ' Year of release: ' || sal\_record.year\_of\_release || CHR(10);

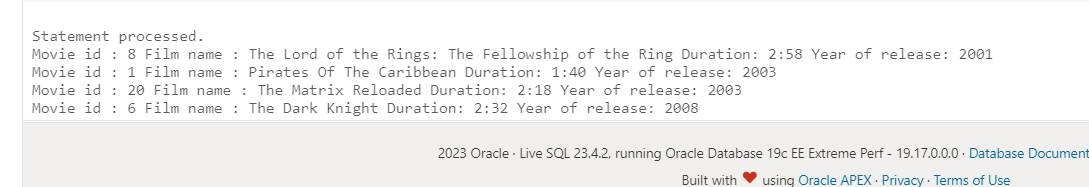
end loop;

return movie\_info;

end;

end mov\_pkg;

execute mov\_pkg.print\_movies



create or replace package genre\_pkg is

procedure fetch\_genre\_movies (my\_genre genre.genre\_name%type );

function get\_genre\_movies(my\_genre genre.genre\_name%TYPE) return varchar2;

end genre\_pkg;

create or replace package body genre\_pkg is

procedure fetch\_genre\_movies (my\_genre in genre.genre\_name%type ) IS

cursor genre\_cursor is select genre\_id, genre\_name from genre where genre\_name = my\_genre;

cursor movie\_g\_info(g\_id NUMBER) IS

select m.movie\_id, m.film\_name, m.year\_of\_release from movie m join movie\_genre mg on mg.movie\_id = m.movie\_id where mg.genre\_id = g\_id;

genre\_rec genre\_cursor%ROWTYPE;

movie\_genre\_info\_rec movie\_g\_info%ROWTYPE;

begin

for genre\_rec in genre\_cursor loop

DBMS\_OUTPUT.PUT\_LINE(genre\_rec.genre\_id || ' Genre name: ' || genre\_rec.genre\_name);

for movie\_genre\_info\_rec IN movie\_g\_info(genre\_rec.genre\_id) LOOP

DBMS\_OUTPUT.PUT\_LINE(

'Movie id: ' || movie\_genre\_info\_rec.movie\_id ||

' Film name: ' || movie\_genre\_info\_rec.film\_name ||

' Year of release: ' || movie\_genre\_info\_rec.year\_of\_release

);

end loop;

end loop;

end fetch\_genre\_movies;

function get\_genre\_movies(my\_genre genre.genre\_name%TYPE)

return varchar2

is

genre\_info VARCHAR2(4000);

begin

for genre\_rec in (select genre\_id, genre\_name from genre where genre\_name = my\_genre) loop

genre\_info := genre\_info || genre\_rec.genre\_id || ' Genre name: ' || genre\_rec.genre\_name || CHR(10);

for movie\_genre\_info\_rec in (select m.movie\_id, m.film\_name, m.year\_of\_release from movie m join movie\_genre mg ON mg.movie\_id = m.movie\_id where mg.genre\_id = genre\_rec.genre\_id) loop

genre\_info := genre\_info ||

'Movie id: ' || movie\_genre\_info\_rec.movie\_id ||

' Film name: ' || movie\_genre\_info\_rec.film\_name ||

' Year of release: ' || movie\_genre\_info\_rec.year\_of\_release || CHR(10);

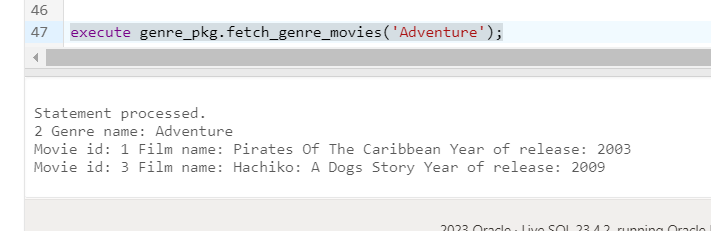
end loop;

end loop;

return genre\_info;

end get\_genre\_movies;

end genre\_pkg;



Questions:

1. Define a function in PL/SQL.
2. Different ways of invoking functions.
3. Packages and their structure.
4. The purpose of creating packages.
5. Private and public components of the package.
6. What is package specification?
7. What is a package body?
8. What is the difference between a procedure declared in the package specification and a procedure declared only in the package body?
9. Explain the notion of the forward declaration.
10. In which cases the forward declaration must be used?
11. What is overloading? And what are its limitations?
12. What is the package state?
13. How is a package created?
14. Is it allowed to initialize variables in the package specification?
15. How are methods invoked inside the package where it is declared?
16. How are methods invoked outside of the package where it is declared?
17. Define the notion of bodiless packages.
18. Describe the BEGIN section in a package.