**Zhumagali Kanat IT2-2003**

**Lab3. Handling exceptions**

1. Add at least three exception handling sections that trap all three different types of exceptions (pre-defined, non-predefined, user-defined).
2. Try incorporating exception handling into the anonymous blocks you created in lab 2.

1.predefined

declare

type fin\_info is record(

m\_id movie.movie\_id%type,

m\_name movie.film\_name%type,

f\_b finance.budget%type,

f\_i finance.income%type

);

f\_rec fin\_info;

begin

select m.movie\_id, m.film\_name, f.budget, f.income into f\_rec from movie\_fin mf join movie m on mf.movie\_id = m.movie\_id join finance f on mf.fin\_id = f.fin\_id;

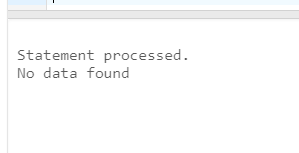
exception

when too\_many\_rows then dbms\_output.put\_line('Too many rows');

when no\_data\_found then dbms\_output.put\_line('No data found');

when others then dbms\_output.put\_line('Some error ocures');

end;



2.non-predifined

declare

inv\_cursor exception;

pragma exception\_init(inv\_cursor, -6511);

cursor dir\_cursor is select director\_id, first\_name, last\_name, date\_of\_birth from directors where extract(year from date\_of\_birth) > 1960;

begin

open dir\_cursor;

for dir\_rec in dir\_cursor loop

dbms\_output.put\_line('ID: '||dir\_rec.director\_id || ' ' ||' First name: ' || dir\_rec.first\_name || ' Last name: ' || dir\_rec.last\_name || ' Date of birth: ' || dir\_rec.date\_of\_birth );

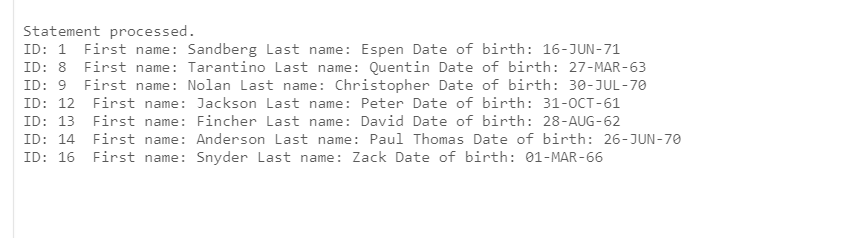
end loop;

close dir\_cursor;

exception

when inv\_cursor then dbms\_output.put\_line(SQLERRM);

end;



3.user-defined

declare

c\_name country.country\_name%type;

data\_col exception;

cursor movie\_cursor is select m.film\_name, c.country\_name from movie\_country mc join movie m on mc.movie\_id = m.movie\_id join country c on mc.country\_id = c.country\_id;

begin

c\_name := 'Japan';

for country\_rec in movie\_cursor loop

if (country\_rec.country\_name = c\_name) then raise data\_col;

else dbms\_output.put\_line(country\_rec.film\_name || ' ' || country\_rec.country\_name);

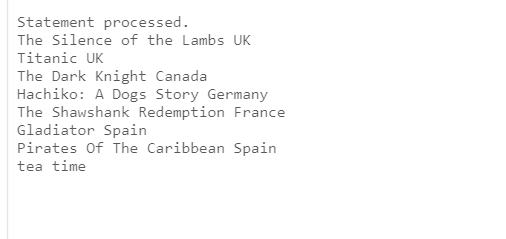
end if;

end loop;

exception

when data\_col then dbms\_output.put\_line('tea time');

end;



sELECT round(dbms\_random.value(1, 20)) FROM dual

declare

m\_id movie.movie\_id%TYPE;

c\_id country.country\_id%TYPE;

begin

FOR i IN 1..20 LOOP

m\_id := i;

c\_id := ROUND(DBMS\_RANDOM.VALUE(1, 20));

insert into movie\_country (movie\_id, country\_id) values (m\_id, c\_id);

commit;

end loop;

end;

lab2 with errors:

declare

yor movie.year\_of\_release%type := 2100;

cursor sal\_cursor is select movie\_id, film\_name, duration, year\_of\_release from movie where year\_of\_release > yor order by year\_of\_release asc;

begin

for sal\_record in sal\_cursor 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;

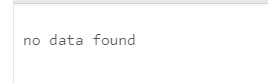
exception

when no\_data\_found then dbms\_output.put\_line('No data found');

when too\_many\_rows then dbms\_output.put\_line('Too many rows');

when others then dbms\_output.put\_line('Some error ocures');

end;



declare

invalid\_id exception;

type movie\_info is record(

movie\_id movie.movie\_id%type,

film\_name movie.film\_name%type,

duration movie.duration%type,

rating\_imbdb movie.rating\_imbdb%type

);

my\_id number;

m\_info movie\_info;

cursor m\_info\_cursor is select movie\_id, film\_name, duration,rating\_imbdb from movie where movie\_id = my\_id;

begin

my\_id := 22;

if my\_id > 20 then raise invalid\_id;

else open m\_info\_cursor;

fetch m\_info\_cursor into m\_info;

close m\_info\_cursor;

dbms\_output.put\_line(m\_info.movie\_id || ' ' || m\_info.film\_name || ' ' || m\_info.duration);

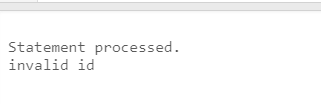
end if;

exception

when invalid\_id then dbms\_output.put\_line('invalid id');

when others then dbms\_output.put\_line(SQLERRM);

end;



declare

budget\_invalid exception;

pragma exception\_init(budget\_invalid, -6502);

type movie\_fin\_info is record(

movie\_id movie.movie\_id%type,

film\_name movie.film\_name%type,

budget finance.budget%type,

income finance.income%type

);

m\_info movie\_fin\_info;

my\_b number;

cursor fin\_cursor is select m.movie\_id, m.film\_name, f.budget, f.income from movie\_fin mf join movie m on mf.movie\_id = m.movie\_id join finance f on mf.fin\_id = f.fin\_id;

begin

my\_b := 100000;

for fin\_record in fin\_cursor loop

m\_info := fin\_record;

if my\_b != to\_number(m\_info.budget) then

raise budget\_invalid;

end if;

if(to\_number(m\_info.budget) > my\_b) then dbms\_output.put\_line('||Film name: ' || m\_info.film\_name || ' has budget: ' || m\_info.budget || ' and his income: ' || m\_info.income || '||');

else dbms\_output.put\_line(m\_info.film\_name || ' has budget whis is less ' || my\_b);

end if;

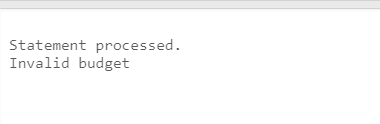
end loop;

exception

when budget\_invalid then dbms\_output.put\_line('Invalid budget ');

when others then dbms\_output.put\_line(SQLERRM);

end;



declare

budget\_invalid exception;

budget\_max\_inv exception;

pragma exception\_init(budget\_invalid, -6502);

type movie\_fin\_info is record(

movie\_id movie.movie\_id%type,

film\_name movie.film\_name%type,

budget finance.budget%type,

income finance.income%type

);

m\_info movie\_fin\_info;

my\_b number;

cursor fin\_cursor is select m.movie\_id, m.film\_name, f.budget, f.income from movie\_fin mf join movie m on mf.movie\_id = m.movie\_id join finance f on mf.fin\_id = f.fin\_id;

max\_b finance.budget%type;

begin

select max(to\_number(budget)) into max\_b from movie\_fin mf join movie m on mf.movie\_id = m.movie\_id join finance f on mf.fin\_id = f.fin\_id;

my\_b := 5000000;

if(my\_b > max\_b) then raise budget\_max\_inv;

end if;

for fin\_record in fin\_cursor loop

m\_info := fin\_record;

if(to\_number(m\_info.budget) > my\_b) then dbms\_output.put\_line('||Film name: ' || m\_info.film\_name || ' has budget: ' || m\_info.budget || ' and his income: ' || m\_info.income || '||');

else dbms\_output.put\_line(m\_info.film\_name || ' has budget whis is less ' || my\_b);

end if;

end loop;

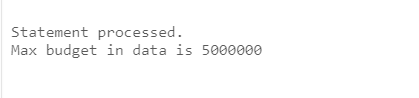
exception

when budget\_max\_inv then dbms\_output.put\_line ('Max budget in data is '|| my\_b);

when budget\_invalid then dbms\_output.put\_line('Invalid budget ');

when others then dbms\_output.put\_line(SQLERRM);

end;



Questions:

1. What is an exception?
2. What types of exceptions exist in PL/SQL?
3. How are exceptions handled?
4. Describe the structure of the exception handling section.
5. What are predefined Oracle server errors and how are they handled?
6. What are non-predefined Oracle server errors and how are they handled?
7. What are user-defined exceptions and how are they handled?