LIRIKA SOLA

TRAVEL AGENCY

REPORT

**TRAVEL AGENCY DATABASE**

**PRESENTATION**

This application is created to help a travel agency organize in a more efficient way the reservations of their excursions on different destinations and time frames. The company covers 5 excursion categories which may be set up in 20 locations all of which have 5 transportation means and 4 accommodation types from where the customer may choose from. Each category has a specific duration, and each location has a fixed price per day. In order for the information to be more organized up to 8 tables were used, every one of which serves a specific purpose and help by making the program less heavy on the memory than it would have if it were unorganized in one table. Furthermore, this way of organization helps the completion of tasks much quicker, satisfying the needs of both staff and owners of the company and contributing in the improvement of decision making.

**TABLE DESCRIPTION**

As aforementioned, the application contains 8 tables, all of which contain a selected category of information and are connected in an efficient way to one another.

The first table is that of the **Employees312**.

1. Employees312 (**Empl\_Code**, Title, Full\_Name, Hire date, Salary, Manager)

The employees are responsible for coordinating the activities of one or more Excursions organized. Each of them has their personal information recorded in the database to keep track of their work and number of projects they take up and manage to complete successfully.

-Empl\_Code is a form of unique ID that identifies the workers from one another.  
-Title contains information about the gender of the employees.  
-Full\_Name contains the employees’ first and last names.  
-Hiredate contains the date on which the employees were hired.  
-Salary contains the salary of each employee without including the commission rate.  
-Manager contains the ID of the managers responsible for one or more employees.

The second table is that of the **Customers312**.

2-Customers312 (**Customer\_Code**, Gender, Title, Age, Full\_Name, Address, *Transportation\_Mean*)

Customers may make one or many reservations. Each of them may choose the excursion category and location of their travel as well as the transportation mean, accommodation and extra facilities that may come with them.

-Customer\_Code is an unique ID that identifies the customers from one another. If the customer has made more than one reservation through the duration of the agency, his ID will be reused.   
-Gender identifies if the customer is a male or a female.  
-Title is used to address customers in a polite manner according to their gender.  
-Age contains information about the age of the customers. Customers should be of legal age in order to book an excursion trip.  
-Full\_Name contains information about the first and last names of the customers.  
-Address contains information about the address of the customers.  
-Transportation\_Mean serves as a key used to reference the Transportation table.

The third table is that of **Categories312**.

Each category focuses on one main activity on which the agency will be based on to organize the itinerary of the trip.

Categories312 (**Category\_Number**, Type, Category\_Desc, Duration)

-Category\_Number is a unique ID that identifies the categories from one another.  
-Type represents the main activity a category number refers to.  
-Category\_Desc contains a short description of each category type.  
-Duration refers to the number of days a trip lasts.

The fourth table is that of **Locations312.**

Each category has up to 20 locations on which the trips may be organized. According to the chosen city, the daily pay rate may vary.

Locations312 (**Location\_Code**, Location\_Name, Price\_Per\_Day*, Category\_Number)*

-Location\_Code is a unique key that identifies each city from one another.  
-Location\_Name refers to the name of the city the trip will be organized on.  
-Price\_Per\_Day contains the amount the customer will pay daily.  
-Category\_Number serves as a key used to reference the Categories table.

The fifth table is **Excursions312.**

Excursions are coordinated by employees and ordered by the customers. As mentioned previously the customers may book more than one reservation for excursions, being that at once or throughout the year(s).

Excursions312 (**Excursion\_Code**, *Location\_Code, Empl\_Code, Customer\_Code,* Reservation\_Date, Trip\_Reservations)

-Excursion\_Code is a unique ID that identifies the excursions organized from one another.  
-Location\_Code is a reference key to the Locations table.  
-Empl\_Code is a reference key to the Employees table.  
-Customer\_Code is a reference key to the Customers table.  
-Reservation\_Date contains the date when the excursion was booked.  
-Trip\_Reservations refers to the number of reservations made by one client at once (for one excursion).

The sixth table is that of **Transportation312**.

When booking the trip, the customers are given information on the mean of transportation they will be using as well as the option to choose their own transportation class (Economy or Business). Depending on their choice, they might get charged extra for the class that it is not covered by the company.

Transportation312 (**Transportation\_Mean**, Transportation\_Class, Included\_In\_Initial\_Price, Extra\_Price)

-Transportation\_Mean refers to the mean of transport the customers will be using to reach their destination. It may be an airplane, bus, train, ferry or private car.  
-Transportation\_Class ranges between Economy (which is included in the initial price) and Business (which if it is chosen, the customers get charged $30 extra).  
-Included\_In\_Initial\_Price contains information whether the class chosen is charged extra or not.  
-Extra\_Price contains the amount the user may be charged extra.

The seventh table is **Accommodation312**.

Similar to Transportation, when booking the trip the users are informed over the type of accommodation they will have. Additionally, they will be given the option of choosing extra facilities that may not be included in the initial price of the rate paid daily. Depending on their choice, they might get charged extra for the facility that it is not covered by the company.

Accommodation312 (**Accommodation\_Type**, Facilities, Included\_In\_Initial\_Price, Extra\_Price1)

-Accommodation\_Type refers to the type of accommodation the costumers will have during their stay abroad. It may be an apartment, Airbnb, Bed and Breakfast or hotel.  
-Facilities refer to the extra services that may be offered to costumers during their stay. The most usual ones are TV, Mini Bar and Room Service (for which the costumer gets charged $20).  
-Included\_In\_Initial\_Price contains information whether the facility chosen is charged extra or not.  
-Extra\_Price contains the amount the user may be charged extra.

The eighth table, **Reservations312**.

This table is an associative entity created by the many to many relationships between the customers and accommodation types (a costumer may choose to reserve more than one apartment/hotel room/Airbnb/Bed and Breakfast at once).

Reservations312 (**Customer\_Code, Accommodation\_Type, Reservation\_Number**)

All the fields of this table are considered primary keys.

-Customer\_Code is a reference key to the Customers table.  
-Accommodation\_Type is a reference key to the Accommodation table.  
-Reservation\_Number is an unique key that represents the accommodation number a customer may have booked.

For a better understanding of the existing connections, an ERD was created (stored in the project file).

**QUERIES**

select \* from customers312;

select \* from employees312;

select \* from locations312;

select \* from categories312;

select \* from excursions312;

select \* from transportation312;

select \* from accommodation312;

select \* from reservations312;

*Q1--LIST RECORDS IN HIERARCHICAL ORDER*

SELECT LEVEL, LPAD(' ',2\*(LEVEL - 1), '\*') || full\_name "Employee",

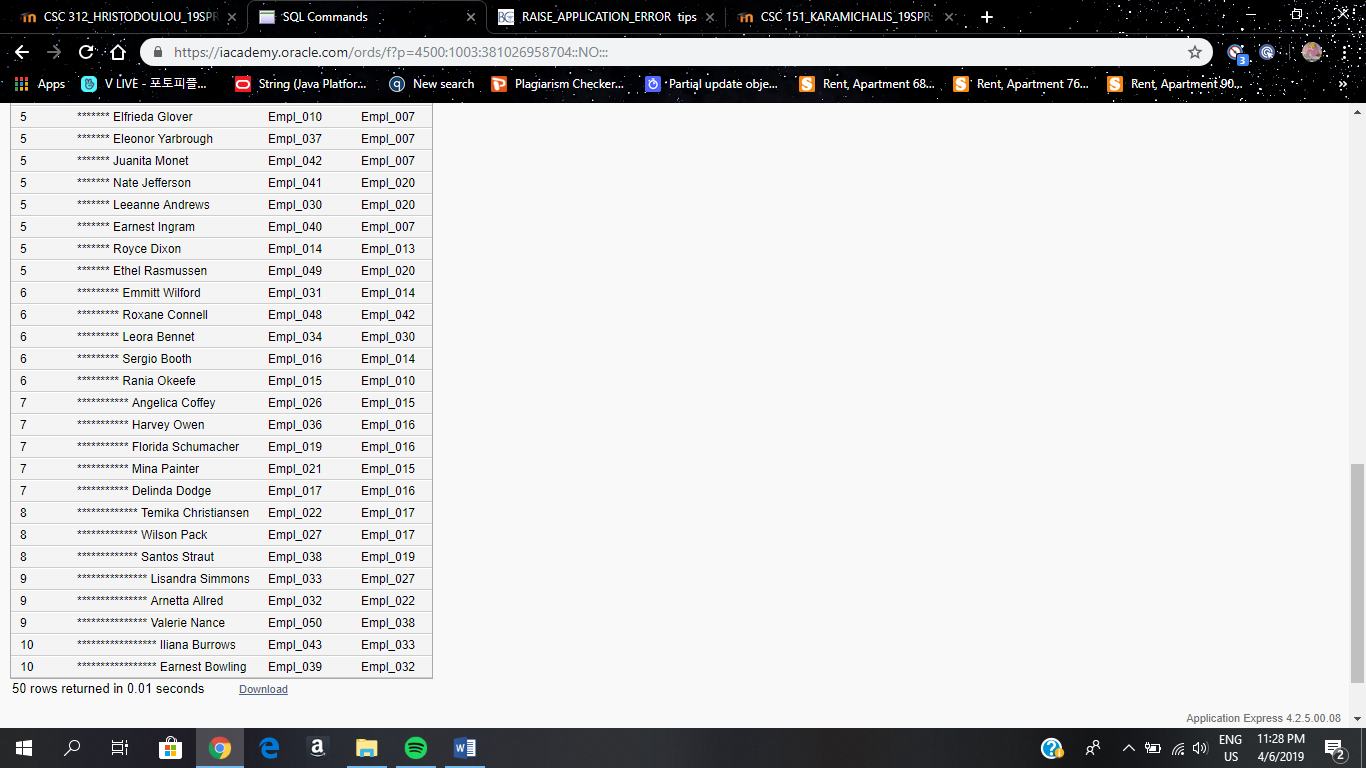
empl\_code, manager

FROM employees312

START WITH manager IS NULL

CONNECT BY PRIOR empl\_code = manager

ORDER BY 1;



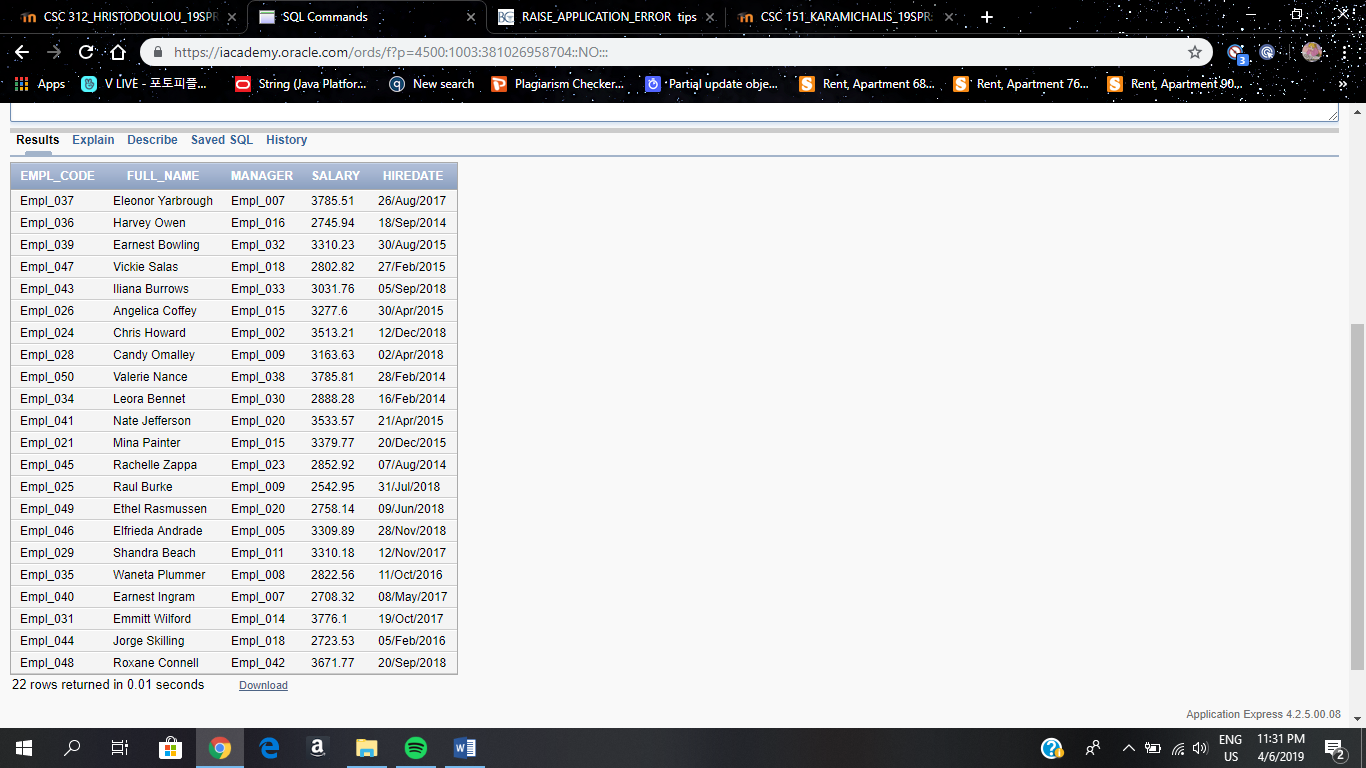
*Q2--DISPLAY EMPLOYEE ID, FULL NAME, MANAGER ID, SALARY AND HIREDATE OF EMPLOYEES WHO DO NOT MANAGE ANYONE BY USING A CORRELATED SUBQUERY*

SELECT empl\_code, full\_name, manager, salary, hiredate

FROM employees312 e

WHERE NOT EXISTS

(SELECT empl\_code FROM employees312 e1 WHERE e.empl\_code = e1.manager);



*Q3--DISPLAY EMPLOYEE CODE, FULL NAME, SALARY, MANAGER, HIREDATE AND NUMBER OF EXCURSIONS, LED BY EMPLOYEES WHO WERE HIRED BEFORE THEIR MANAGERS ON THE FOURTH DAY OF THE MONTH.*

select a.Empl\_Code, a.Full\_Name, a.Salary, a.Manager, a.Hiredate, (select count(ex.Excursion\_Code) from Excursions312 ex where a.Empl\_Code=ex.Empl\_Code)

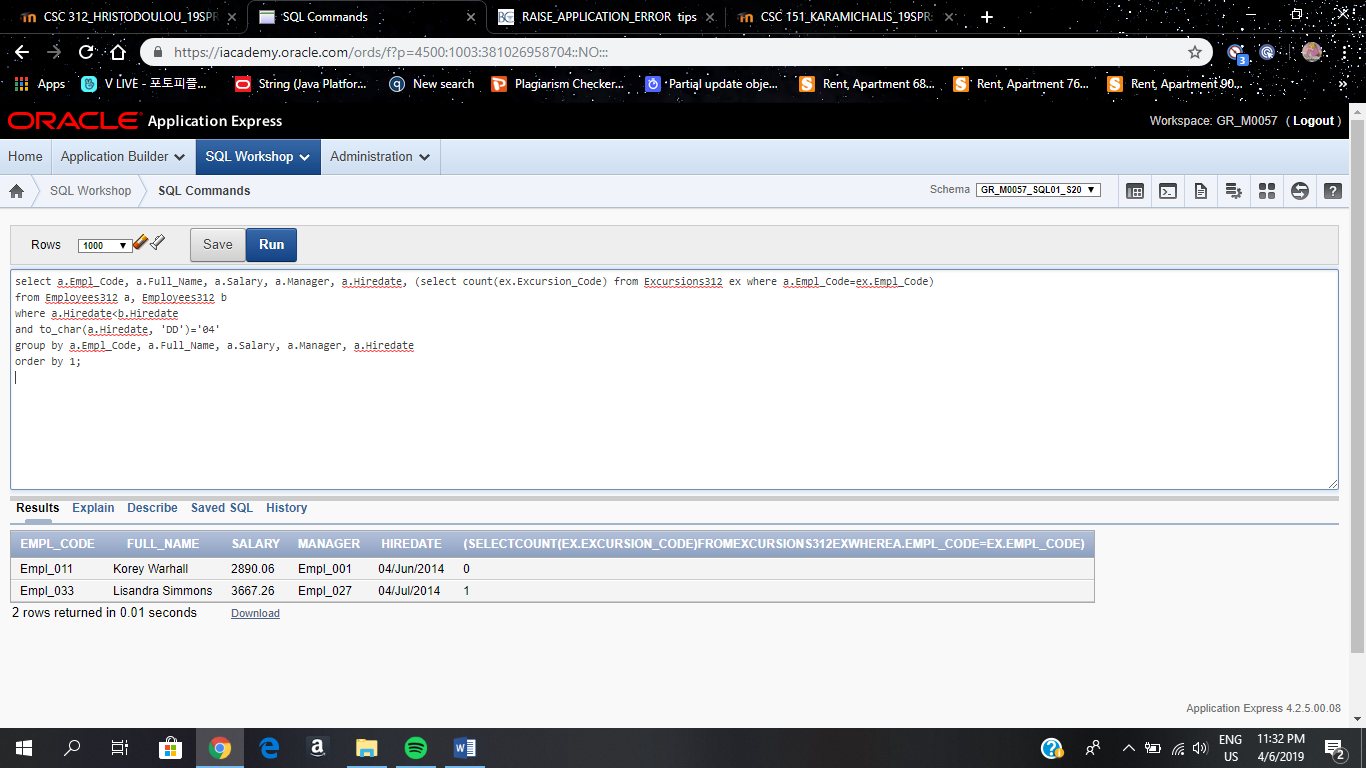
from Employees312 a, Employees312 b

where a.Hiredate<b.Hiredate

and to\_char(a.Hiredate, 'DD')='04'

group by a.Empl\_Code, a.Full\_Name, a.Salary, a.Manager, a.Hiredate

order by 1;



*Q5--DISPLAY EMPLOYEE'S CODE, FULL NAME AND THE CODE OF THE EXCURSIONS THEY LED, WHO HAD MORE THAN 8 RESERVATIONS MADE UNDER.*

select e.empl\_code, e.full\_name, dd.excursion\_code

from employees312 e join

(

select ex.empl\_code, ex.excursion\_code, ex.trip\_reservations

from employees312 e join excursions312 ex

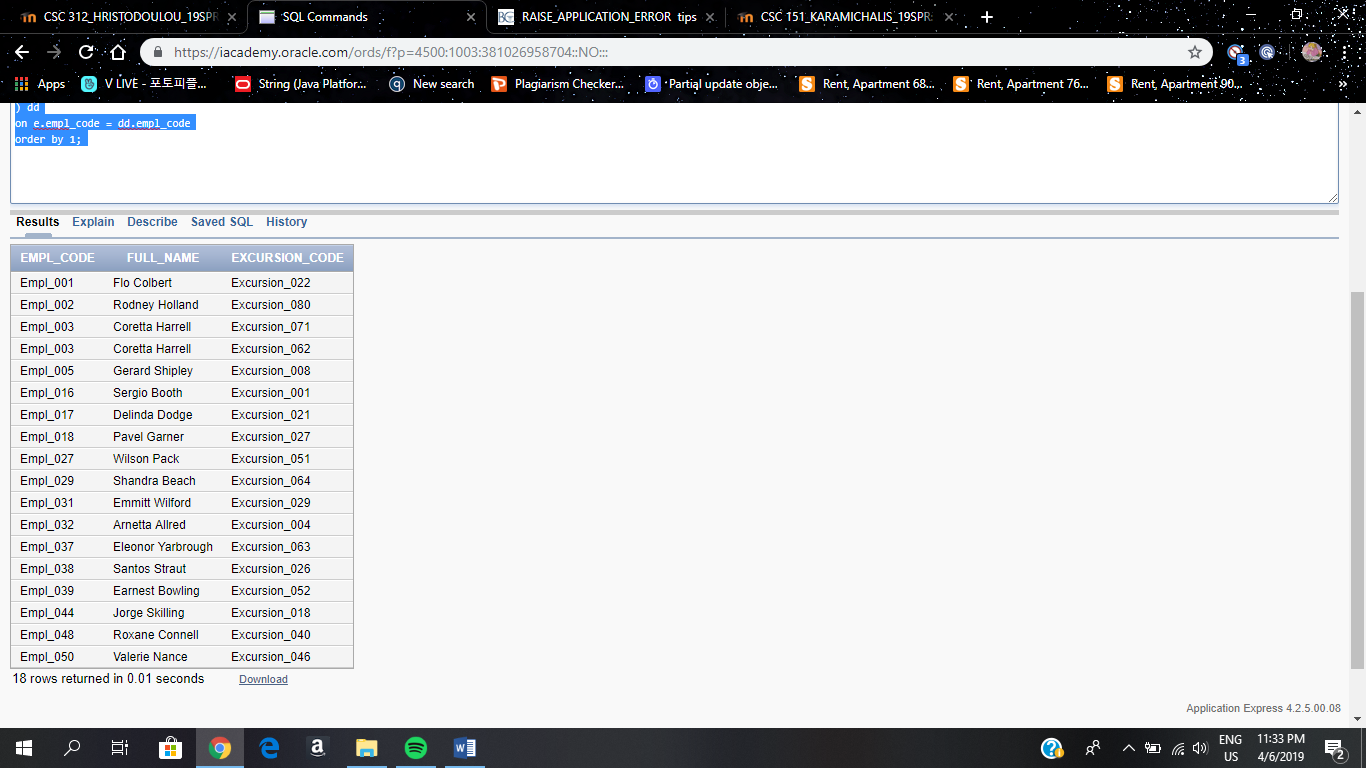
on e.empl\_code = ex.empl\_code

where ex.trip\_reservations>8

) dd

on e.empl\_code = dd.empl\_code

order by 1;



*Q6--PRODUCE A REPORT WITH THE FOLLOWING STRUCTURE: CUSTOMER CODE, TITLE, FULL NAME, TRANSPORTATION MEAN, TRANSPORTATION CLASS, NUMBER OF RESERVED EXCURSIONS MADE IN 2016, 2017, 2018.*

*THE REPORT SHOULD CONTAIN INFORMATION ONLY FOR CUSTOMERS WITH ID HIGHER THAN \_005 AND LOWER THAN \_025, WHO ARE FEMALES.*

*TRANSPORTATION CLASS SHOULD HAVE THE FOLLOWING STRUCTURE:*

*Economy\_Class 'Included in the initial price.'*

*Business\_Class 'Charged 30$ extra.'*

*'Unspecified'*

***Note****: If some information is missing, you should replace nulls with appropriate values.*

drop table temporary\_customer;

select \* from temporary\_customer;

create table temporary\_customer as

(

select c.customer\_code, c.title, c.full\_name,

c.transportation\_mean,

case t.transportation\_class

when 'Economy\_Class' then 'Included in the initial price.'

when 'Business\_Class' then 'Charged 30$ extra.'

else 'Unspecified'

end trans\_class

from customers312 c join transportation312 t

on c.transportation\_mean=t.transportation\_mean

where c.customer\_code between 'Cust\_005' and 'Cust\_025' and

c.gender='Female');

drop table temp\_counting;

select \* from temp\_counting;

create table temp\_counting as

(

select customer\_code,

sum(decode(to\_char(reservation\_date, 'YYYY'), '2016', 1,0)) counting\_2016,

sum(decode(to\_char(reservation\_date, 'YYYY'), '2017', 1,0)) counting\_2017,

sum(decode(to\_char(reservation\_date, 'YYYY'), '2018', 1,0)) counting\_2018

from excursions312

group by customer\_code

);

select c.customer\_code, c.title, c.full\_name, c.transportation\_mean, c.trans\_class,

nvl(t.counting\_2016,0) "Reserved Excursions 2016",

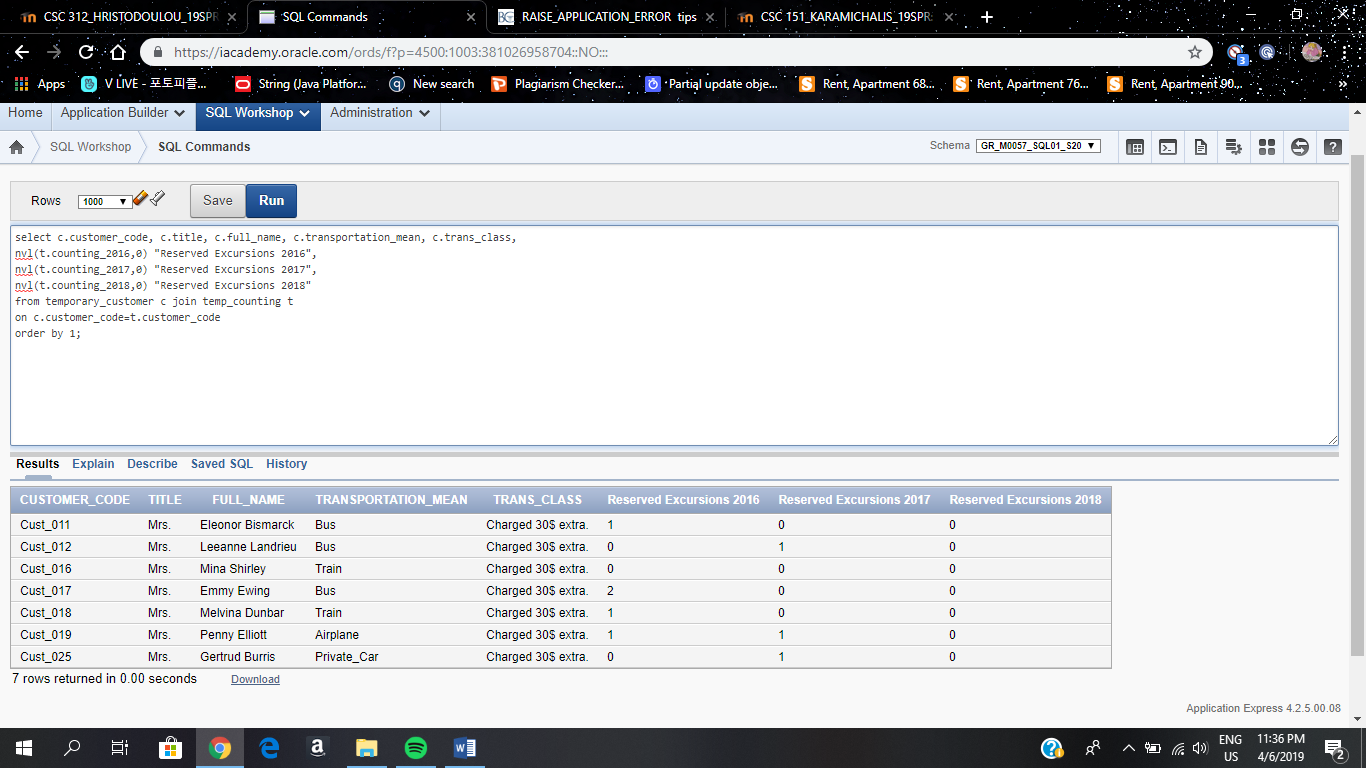
nvl(t.counting\_2017,0) "Reserved Excursions 2017",

nvl(t.counting\_2018,0) "Reserved Excursions 2018"

from temporary\_customer c join temp\_counting t

on c.customer\_code=t.customer\_code

order by 1;



*Q7--CREATE A FUNCTION THAT RETURNS THE REVENUE GENERATED BY THE EXCURSIONS ORDERED IN EACH CATEGORY.*

--VIEW

select \* from Excursions\_v1;

create or replace view Excursions\_v1 as

select b.Category\_Number, b.Type, sum(((substr(b.Duration,1,1)\*c.Price\_Per\_Day\*d.Trip\_Reservations)+f.Extra\_Price+h.Extra\_Price1)) as "REVENUE"

from Categories312 b join Locations312 c on b.Category\_Number=c.Category\_Number join Excursions312 d on c.Location\_Code=d.Location\_Code

join Customers312 e on d.Customer\_Code=e.Customer\_Code join Transportation312 f on e.Transportation\_Mean=f.Transportation\_Mean

join Reservations312 g on e.Customer\_Code=g.Customer\_Code join Accommodation312 h on g.Accommodation\_Type=h.Accommodation\_Type

group by b.Category\_Number,b.Type

order by 3 desc;

--FUNCTION

create or replace function Revenue\_Check(categ excursions\_v1.category\_number%type)

return Excursions\_v1.revenue%type is

v\_rev Excursions\_v1.revenue%type;

begin

select revenue

into v\_rev

from Excursions\_v1

where category\_number=categ;

return(v\_rev);

end;

--RETRIEVE THE FUNCTION

declare

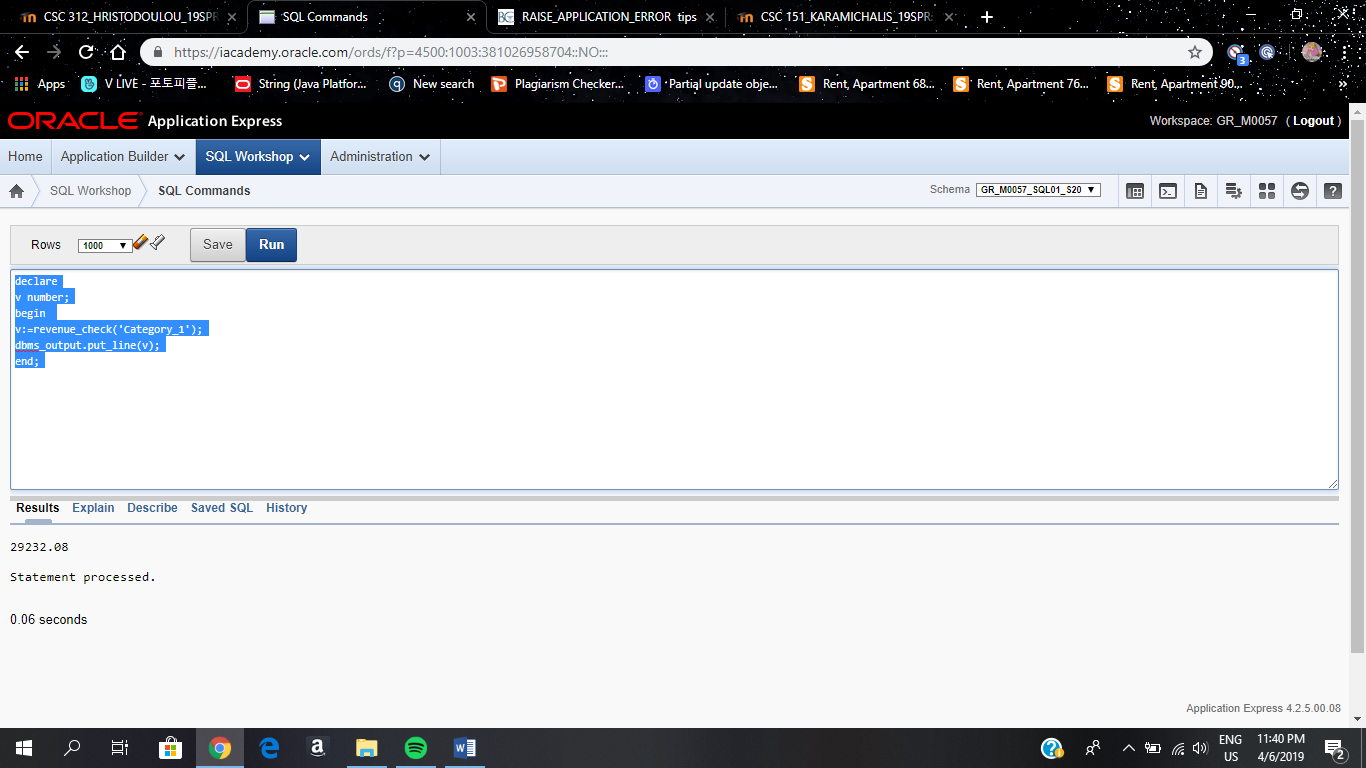
v number;

begin

v:=revenue\_check('Category\_1');

dbms\_output.put\_line(v);

end;



*Q8--ASSUMING THAT THE COMMISSION RATE OF EMPLOYEES IS 14%, CREATE A PROCEDURE THAT CALCULATES THE TOTAL COMMISSION RATE OF EMPLOYEES PER EACH CATEGORY.*

CREATE OR REPLACE PROCEDURE CAT\_COMMISSION(categ excursions\_v1.category\_number%type)

is

v\_com Excursions\_v1.revenue%type;

begin

select revenue\*0.14

into v\_com

from Excursions\_v1

where category\_number=categ;

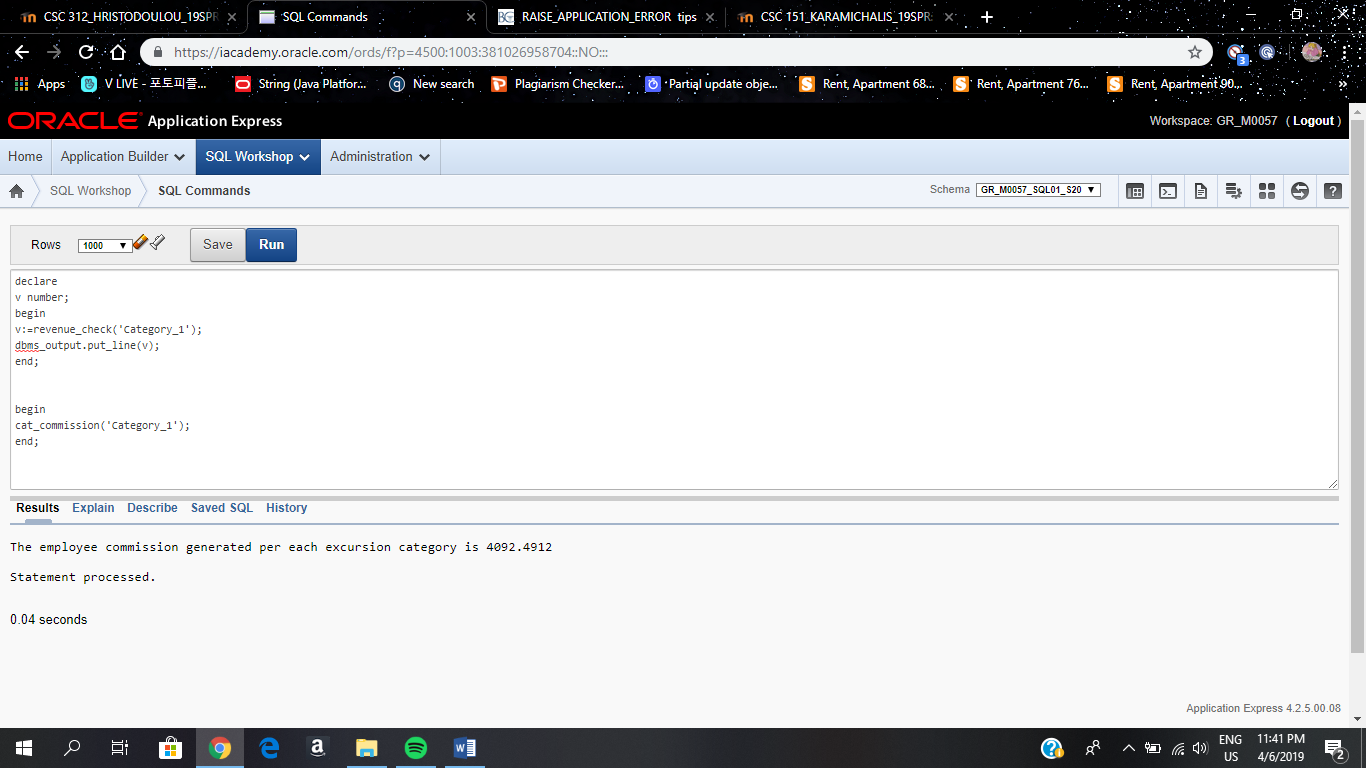
dbms\_output.put\_line('The employee commission generated per each excursion category is ' || v\_com);

end;

begin

cat\_commission('Category\_1');

end;



Q9--CREATE A FUNCTION THAT RETURNS THE CATEGORY OF THE EXCURSION WHEN GIVEN AS ARGUMENTS THE EXCURSION ID AND LOCATION WHERE IT TOOK PLACE

create or replace function getCategory(ex\_id excursions312.excursion\_code%type, loc\_code excursions312.location\_code%type)

return locations312.category\_number%type

is

categ varchar2(20);

begin

select category\_number

into categ

from locations312 l join excursions312 ex

on l.location\_code= ex.location\_code

where ex.excursion\_code=ex\_id and ex.location\_code=loc\_code;

return(categ);

end;

--RETRIEVE IT

declare

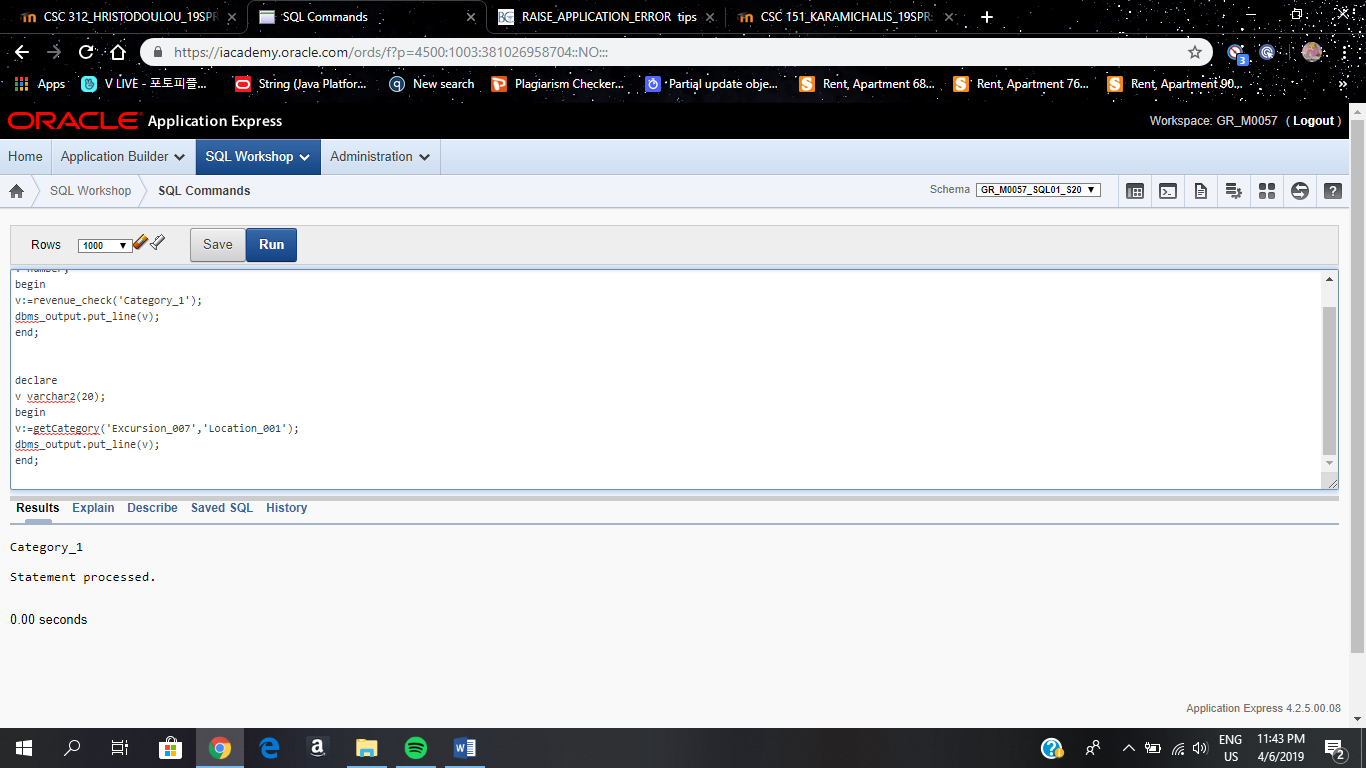
v varchar2(20);

begin

v:=getCategory('Excursion\_007','Location\_001');

dbms\_output.put\_line(v);

end;



*Q10--CREATE A TRIGGER THAT WHEN INSERTING ACCOMMODATION TYPES AND FACILITIES IN THE ACCOMMODATION312 TABLE WILL SUBSTITUTE POTENTIAL SPACES WITH '-'.*

*IF THE USER DECIDES TO UPDATE THE FIELDS IN THIS TABLE AN ERROR MESSAGE SHOULD BE DISPLAYED TO HIM.*

CREATE OR REPLACE TRIGGER accommodation\_insert

BEFORE INSERT OR UPDATE OF accommodation\_type, facilities ON accommodation312

FOR EACH ROW

WHEN (REGEXP\_LIKE(new.accommodation\_type,' ') or REGEXP\_LIKE(new.facilities,' '))

BEGIN

IF INSERTING THEN

:new.accommodation\_type :=

REGEXP\_REPLACE(:new.accommodation\_type,' ','-',1);

:new.facilities:=

REGEXP\_REPLACE(:new.facilities,' ','-',1);

ELSIF UPDATING THEN

IF REGEXP\_LIKE(:new.accommodation\_type,' ') THEN

RAISE\_APPLICATION\_ERROR(-20099,'Updates can''t use multiple names for accommodation types.');

ELSIF REGEXP\_LIKE(:new.facilities,' ') THEN

RAISE\_APPLICATION\_ERROR(-20099,'Updates can''t use multiple names for facility types.');

END IF;

END IF;

END accommodation\_insert;

*--INSERT STATEMENTS TO PROVE THE ENFORCEMENT OF THE TRIGGER*

INSERT INTO accommodation312

(accommodation\_type

, facilities

, included\_in\_initial\_price

, extra\_price1

)

VALUES

( 'Cottage Y Y'

, 'House Keeper'

, 'No'

, 20

);

*--UPDATE STATEMENT TO PROVE THE ENFORCEMENT OF THE TRIGGER*

update accommodation312

set accommodation\_type='Cottage Y Y', facilities='House Keeper'

where accommodation\_type='Cottage-Y-Y' and facilities='House-Keeper';

*--DELETE THE ROW INSERTED TO UNDO THE INSERTION*

delete from accommodation312

where accommodation\_type='Cottage-Y-Y' and facilities='House-Keeper';

Q11--USE A CURSOR TO RETRIEVE THE EMPLOYEE CODE AND NAME FROM THE EMPLOYEES312. PASS THE EMPLOYEE CODE TO ANOTHER CURSOR TO RETRIEVE FROM THE EXCURSIONS312 TABLE

THE DETAILS OF THE EXCURSIONS THAT THEY HAVE MANAGED.

DECLARE

CURSOR C\_EMPL IS

SELECT empl\_code, full\_name

FROM employees312;

lv\_empl employees312.empl\_code%TYPE;

lv\_name employees312.full\_name%TYPE;

CURSOR C\_EXC IS

SELECT excursion\_code, reservation\_date

FROM excursions312

WHERE empl\_code= lv\_empl

ORDER BY excursion\_code;

lv\_exc\_code excursions312.excursion\_code%TYPE;

lv\_res excursions312.reservation\_date%TYPE;

BEGIN

OPEN c\_empl;

LOOP

FETCH c\_empl INTO lv\_empl, lv\_name;

IF c\_empl%NOTFOUND

THEN

EXIT;

END IF;

Dbms\_output.put\_line('Employee Code: '||lv\_empl||' Employee Name : '||lv\_name);

OPEN C\_EXC;

LOOP

FETCH C\_EXC INTO lv\_exc\_code, lv\_res;

IF C\_EXC%NOTFOUND

THEN

EXIT;

END IF;

Dbms\_output.put\_line(' Excursion Code : '||lv\_exc\_code||' Reservation Date : '||lv\_res);

END LOOP;

CLOSE c\_exc;

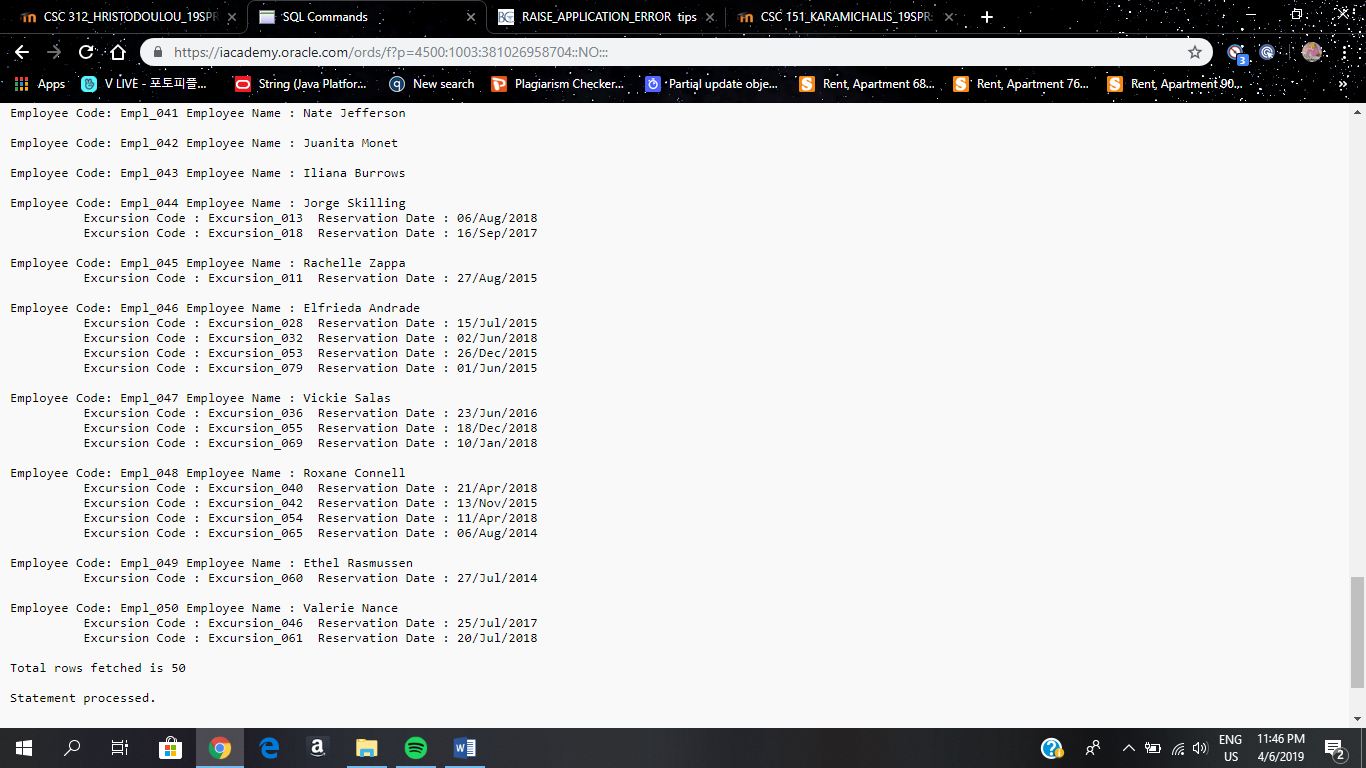
Dbms\_output.put\_line(' ');

END LOOP;

Dbms\_output.put\_line('Total rows fetched is '||c\_empl%ROWCOUNT);

CLOSE c\_empl;

END;

****

**SUGGESTIONS FOR FURTHER IMPROVEMENT**

Since this database was created in a normalized way, I believe that there is not much to change in the logic of the application. However, there is room for improvement when it comes to the expansion of it. A good idea would be to elaborate more on the transportation means by allowing the costumers to choose, similarly to accommodations, their way and class of transportation. Each transportation mean other than the one provided by the company (and included in the initial price) would charge extra to the costumers. Additionally, for the accommodation type, the users would be provided with the opportunity of choosing the area of the city they want to spend their trip on.  
Another field that could have been added to the Customers table would be a Discount (or Membership) one for costumers that have used the agency’s services more than 3 times.   
Similar to that a Coupon or Packet field could have been added to the Excursions table for reservations done during the holiday period. Adding this fields would require additional ones when it comes to further calculating the trip expenses.