

19p0012 Aitzaz tahir ch

DATA BASE LAB 03



**Task 3.1 Suppose the owners of all the theme parks wanted to compare the current ticket**

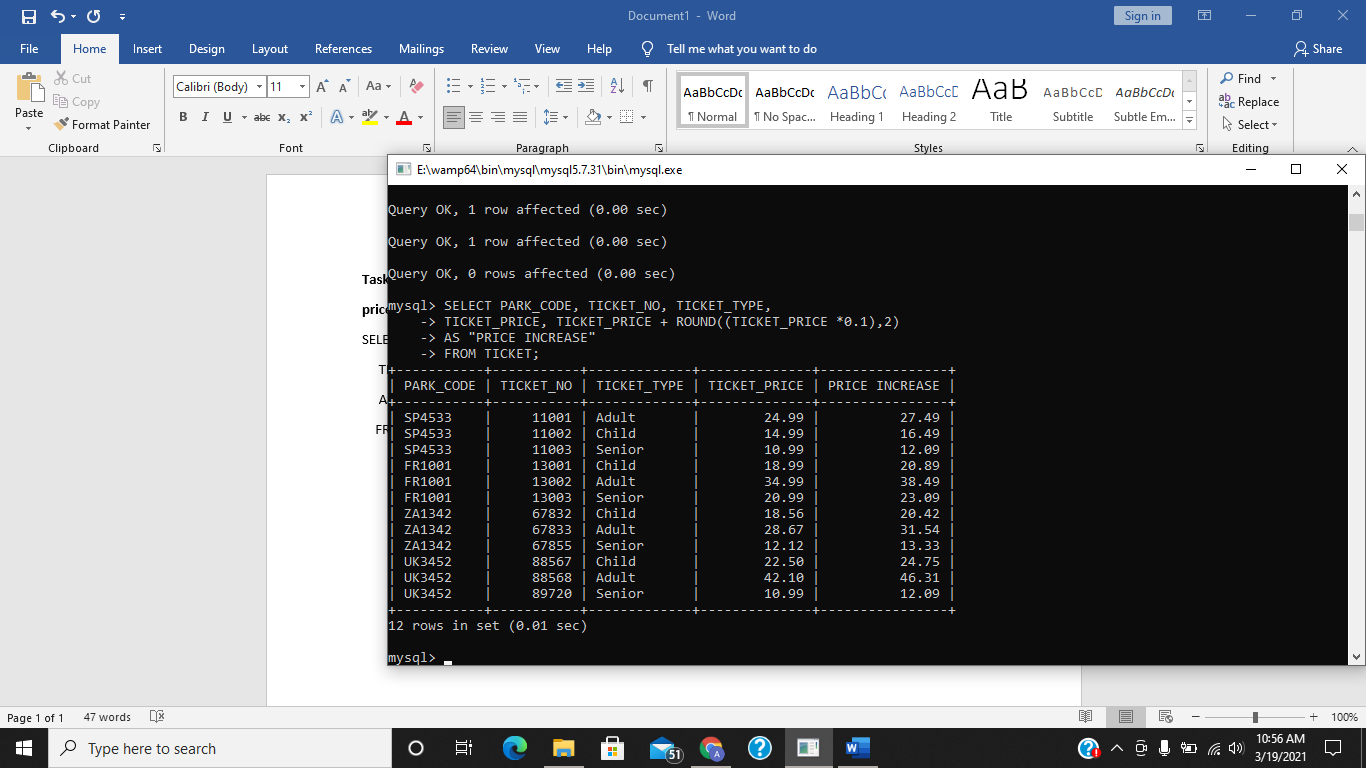
**prices, with an increase in the price of each ticket by 10%. To generate this query type:**

SELECT PARK\_CODE, TICKET\_NO, TICKET\_TYPE,

TICKET\_PRICE, TICKET\_PRICE + ROUND((TICKET\_PRICE \*0.1),2)

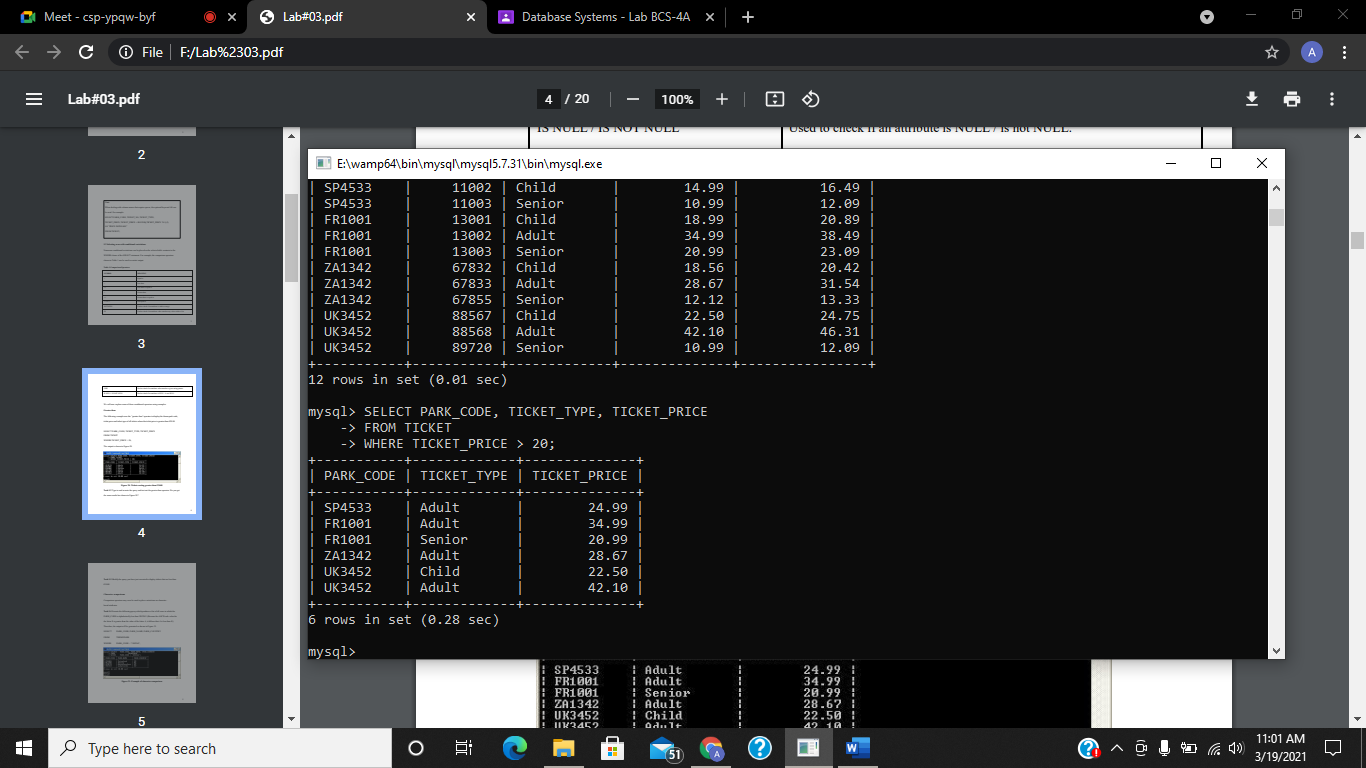
AS "PRICE INCREASE"

FROM TICKET;



**Task 3.2 Type in and execute the query and test out the greater than operator. Do you get the same results has shown in Figure 20?**

Yes as u can see below

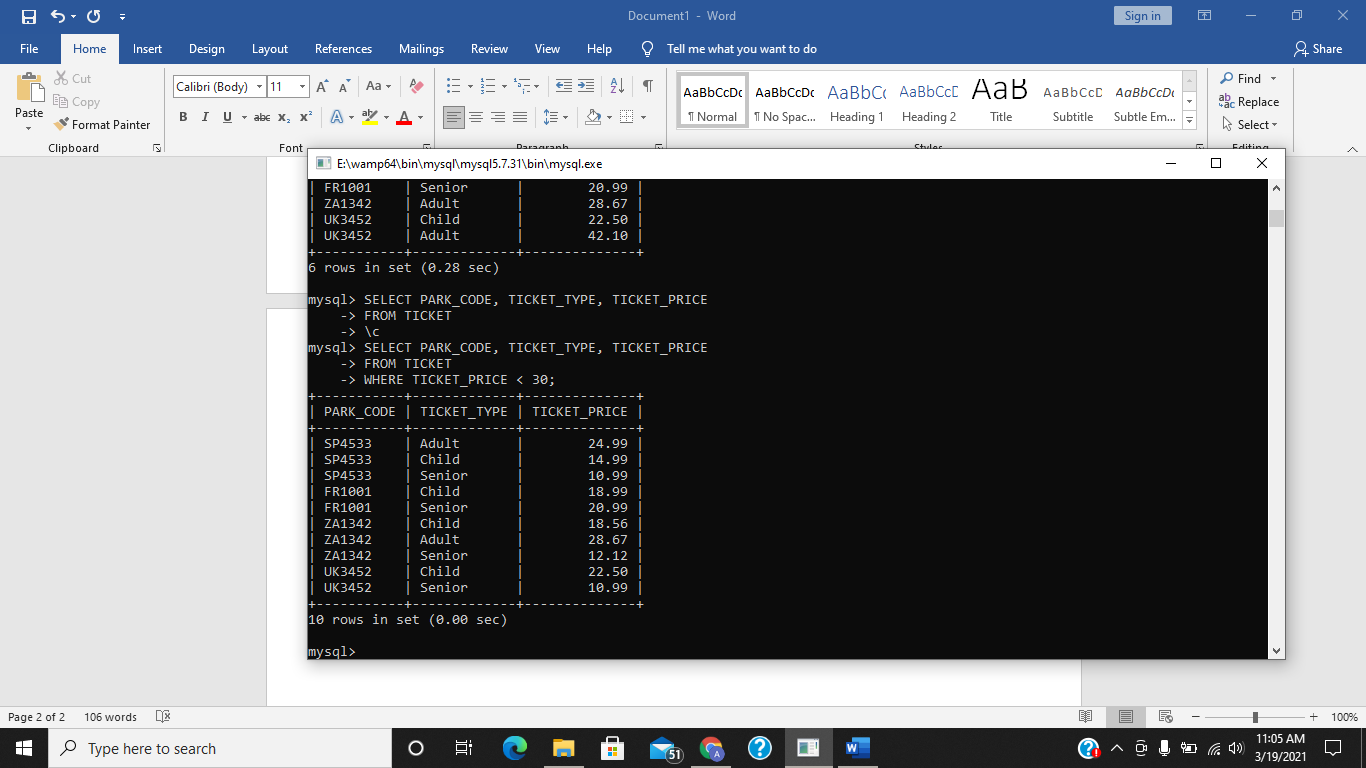


**Task 3.3 Modify the query you have just executed to display tickets that are less than €30.00.**

SELECT PARK\_CODE, TICKET\_TYPE, TICKET\_PRICE

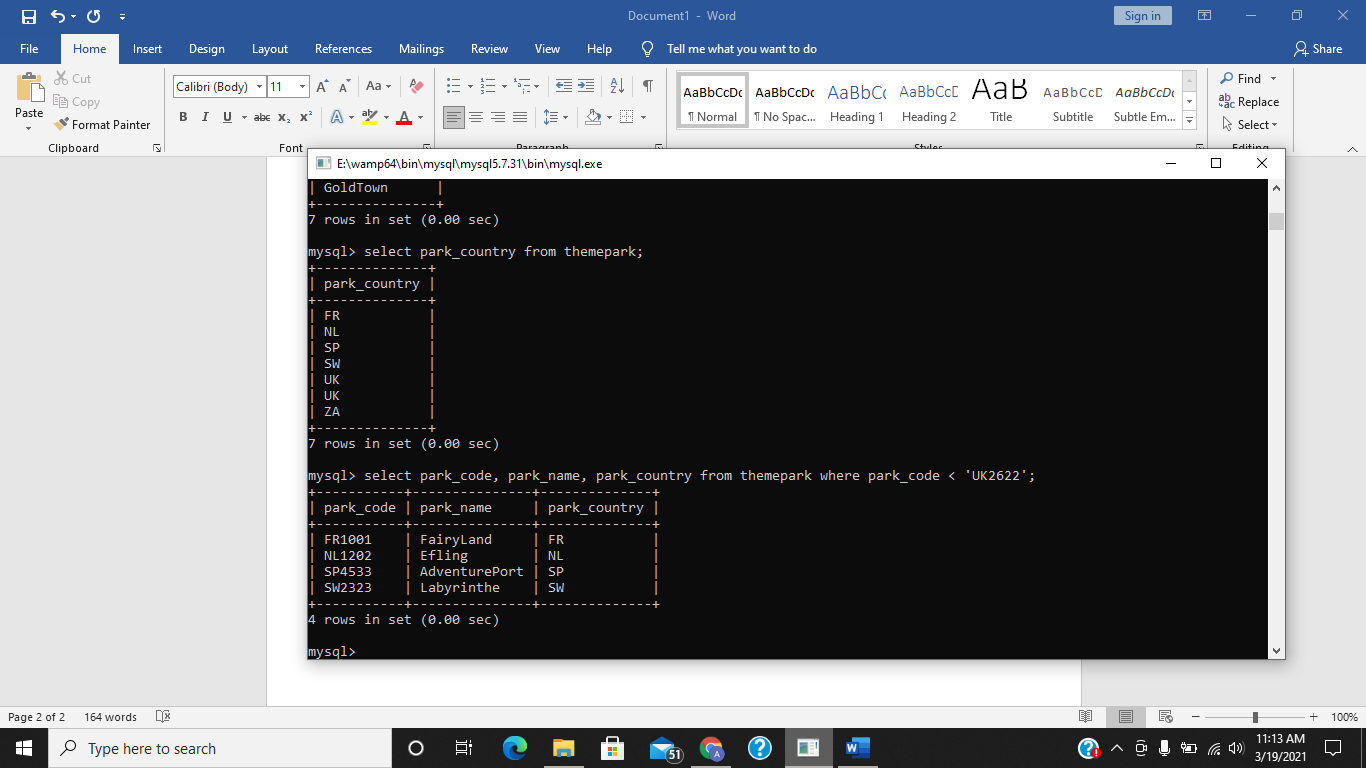
FROM TICKET

WHERE TICKET\_PRICE < 30;



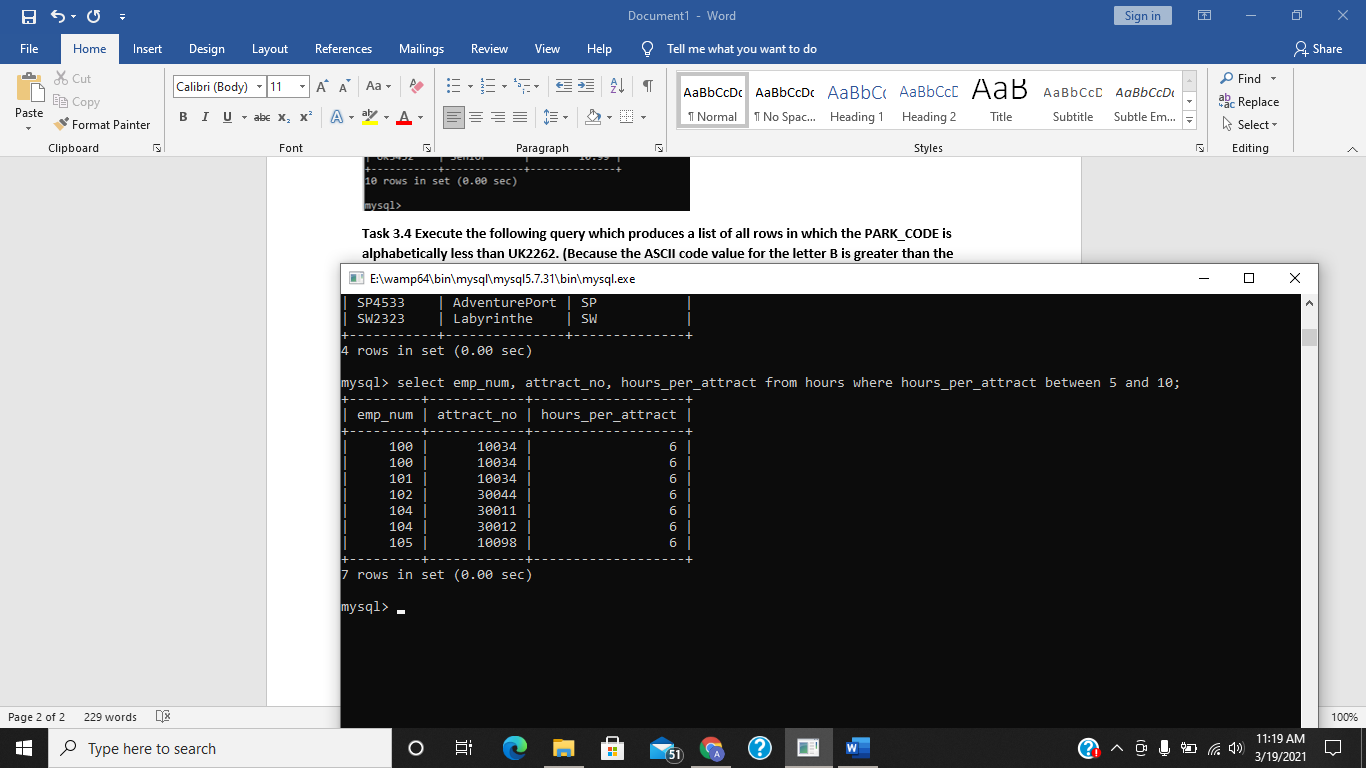
**Task 3.4 Execute the following query which produces a list of all rows in which the PARK\_CODE is alphabetically less than UK2262. (Because the ASCII code value for the letter B is greater than the value of the letter A, it follows that A is less than B.)**

select park\_code, park\_name, park\_country from themepark where park\_code < 'UK2622';



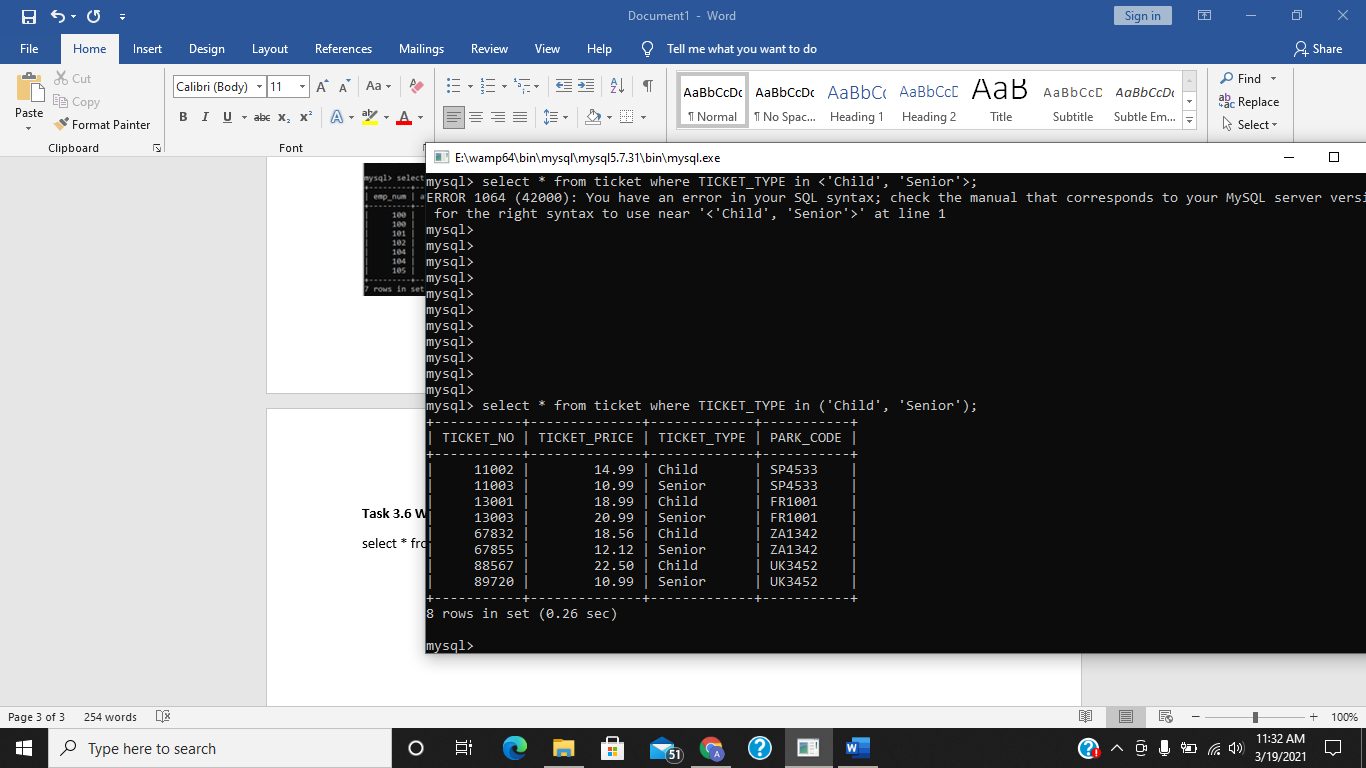
**Task 3.5 Write a query which displays the employee number, attraction no, the hours worked per attraction and the date worked where the hours worked per attraction is between 5 and 10. Hint you will need to select data from the HOURS table. The output for the query is shown in Figure 23.**

select emp\_num, attract\_no, hours\_per\_attract from hours where hours\_per\_attract between 5 and 1;



**Task 3.6 Write a query to display all tickets that are of type Senior or Child.**

select \* from ticket where TICKET\_TYPE in ('Child', 'Senior');

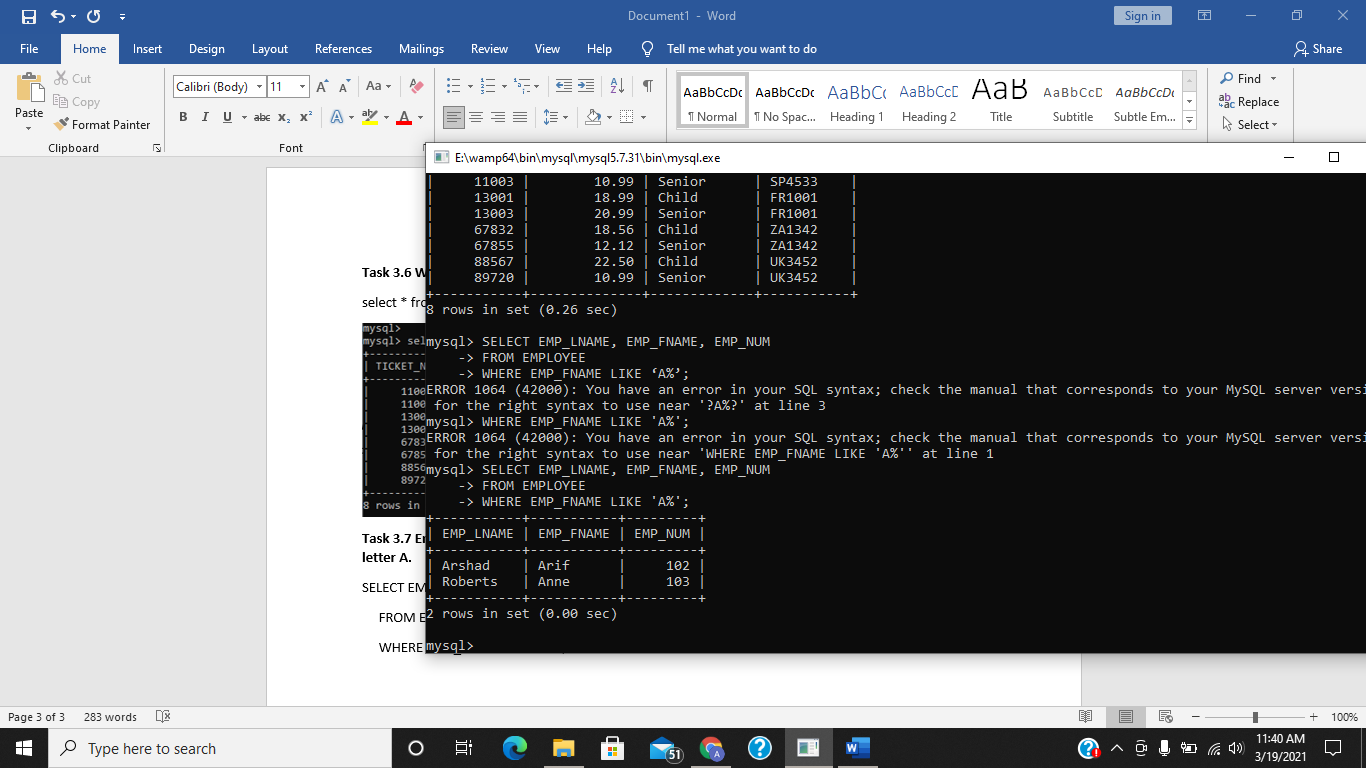


**Task 3.7 Enter the following query which finds all EMPLOYEE rows whose first names begin with the letter A.**

SELECT EMP\_LNAME, EMP\_FNAME, EMP\_NUM

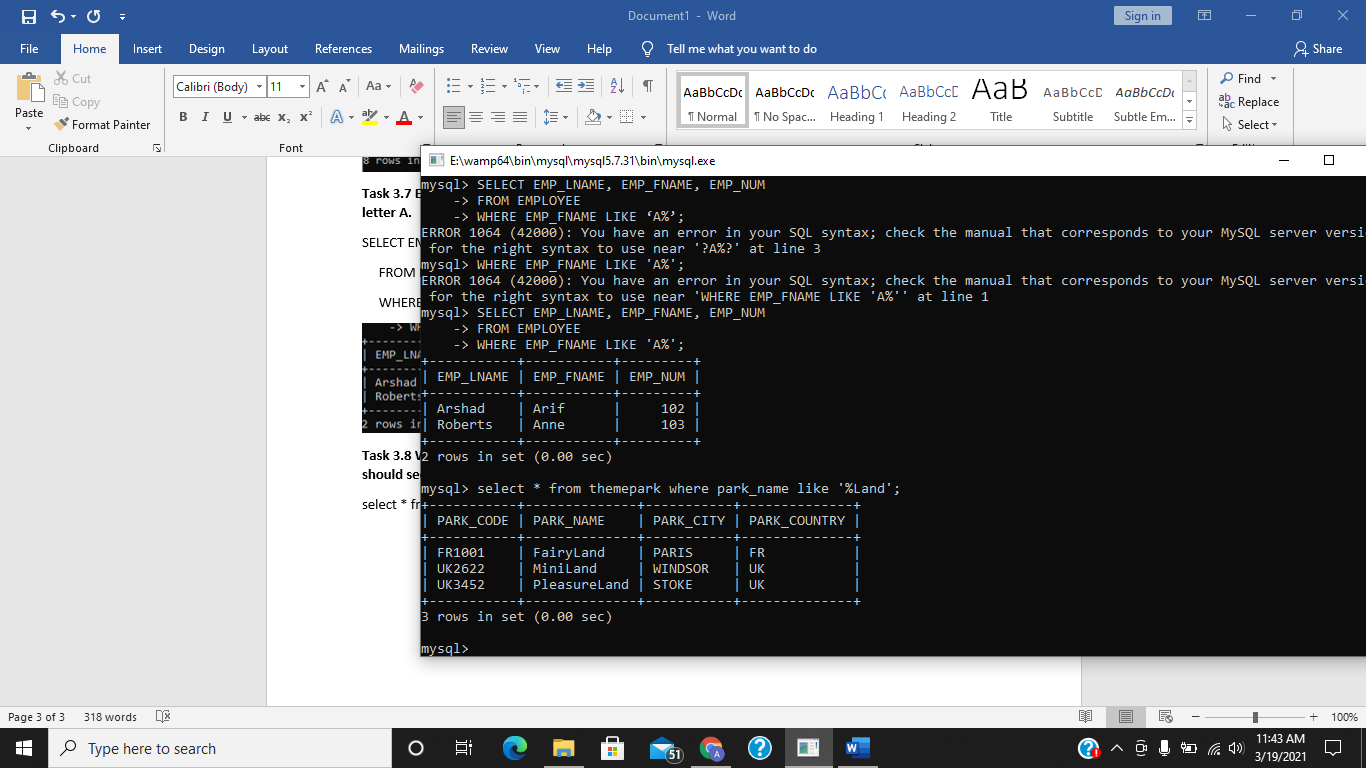
FROM EMPLOYEE

WHERE EMP\_FNAME LIKE 'A%';



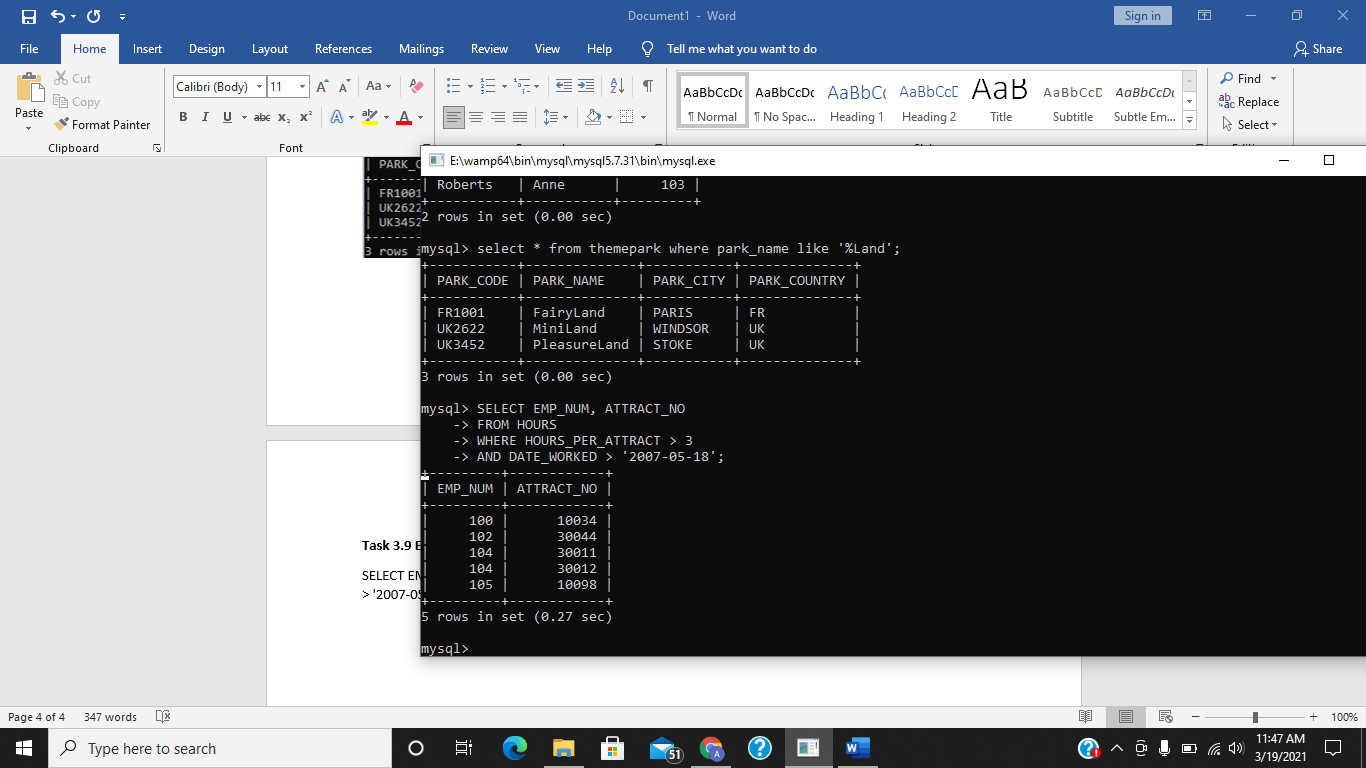
**Task 3.8 Write a query which finds all Theme Parks that have a name ending in ‘Land’. The output you should see is shown in Figure 27.**

select \* from themepark where park\_name like '%Land';



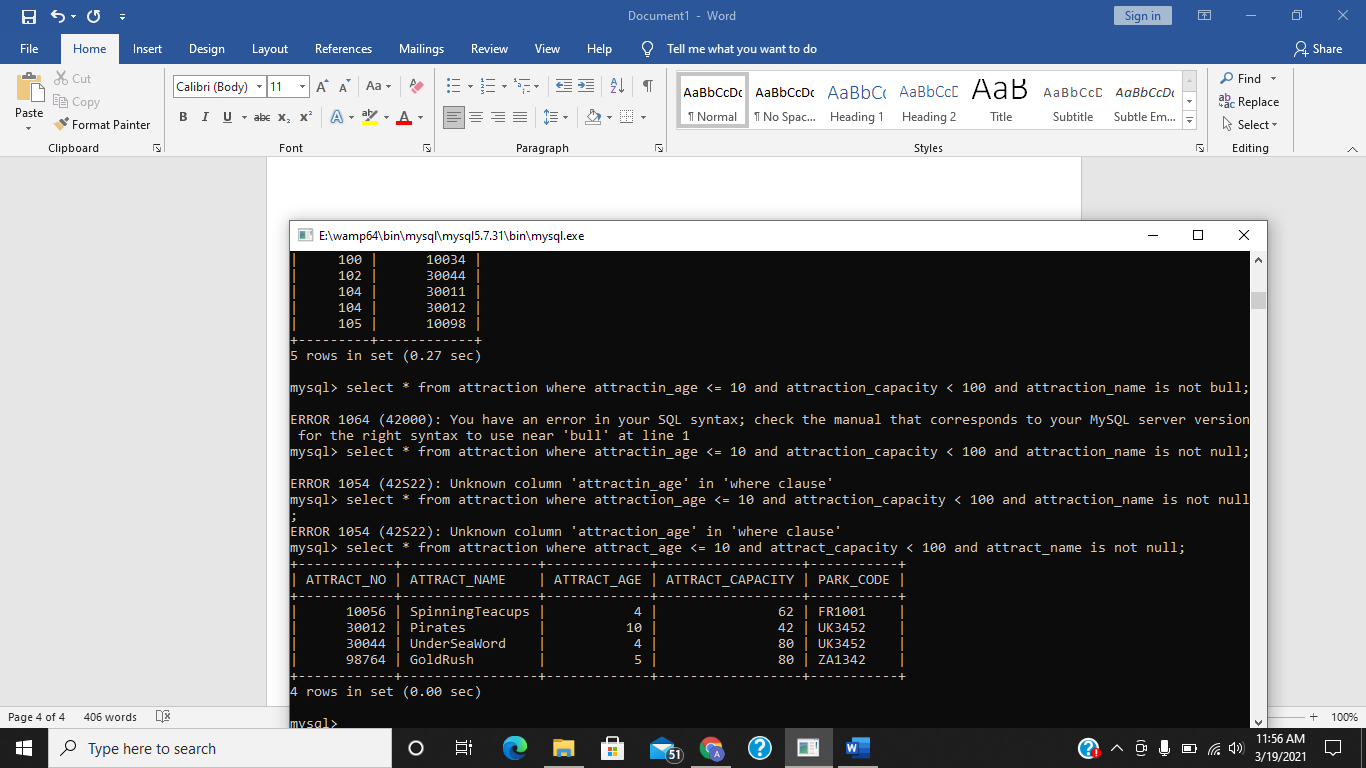
**Task 3.9 Enter the query above and check you results with those shown in Figure 29.**

SELECT EMP\_NUM, ATTRACT\_NO FROM HOURS WHERE HOURS\_PER\_ATTRACT > 3 AND DATE\_WORKED > '2007-05-18';



**Task 3.10 Write a query which displays the details of all attractions which are suitable for children aged 10 or under and have a capacity of less than 100. You should not display any information for attractions which currently have no name.**

select \* from attraction where attract\_age <= 10 and attract\_capacity < 100 and attract\_name is not null;



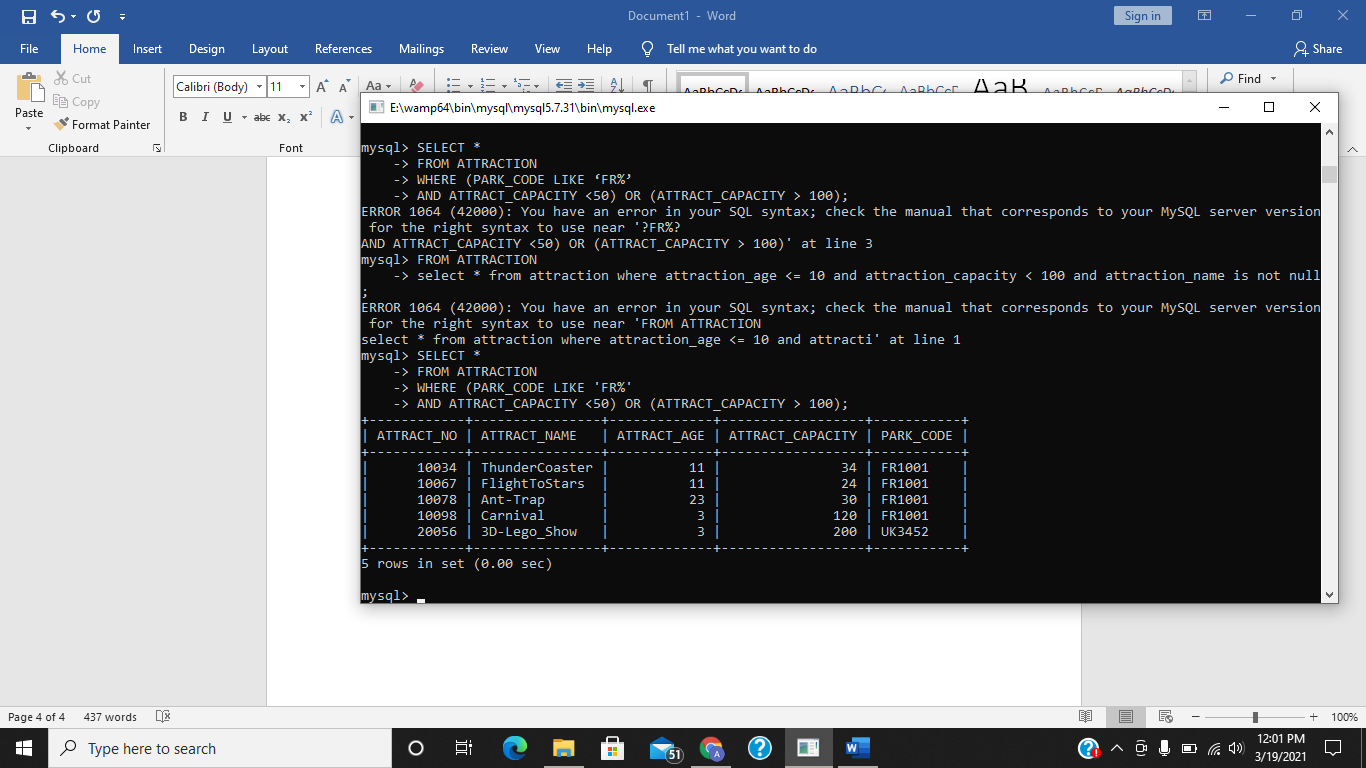
**Task 3.11 Test the following query and check your output with that shown in Figure 32.**

SELECT \*

FROM ATTRACTION

WHERE (PARK\_CODE LIKE 'FR%'

AND ATTRACT\_ CAPACITY < 50) OR (ATTRACT\_CAPACITY > 100);

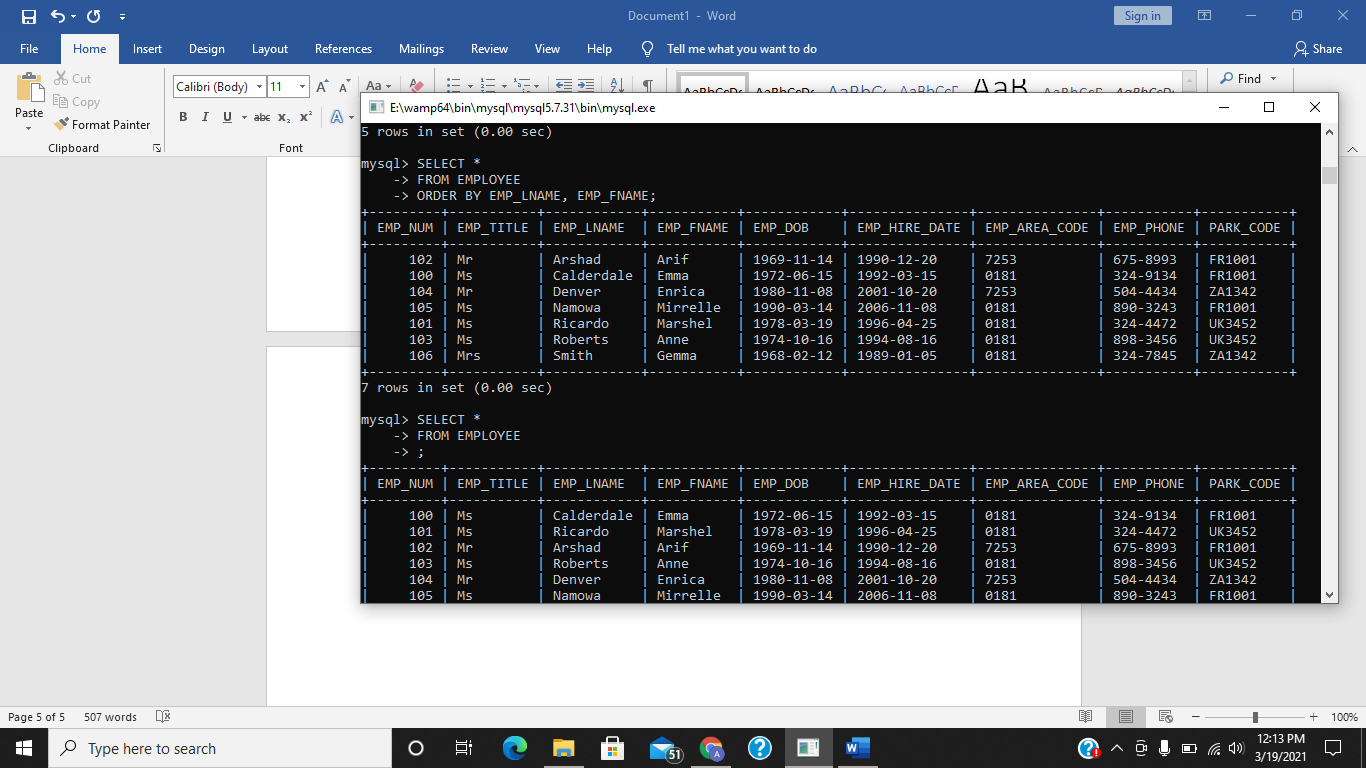


**Can you work out what this query is doing?**

This query is showing us data **PARK\_CODE**  starting with **FR**  and having **ATTRACT\_ CAPACITY** less than 50 or greater than 100.

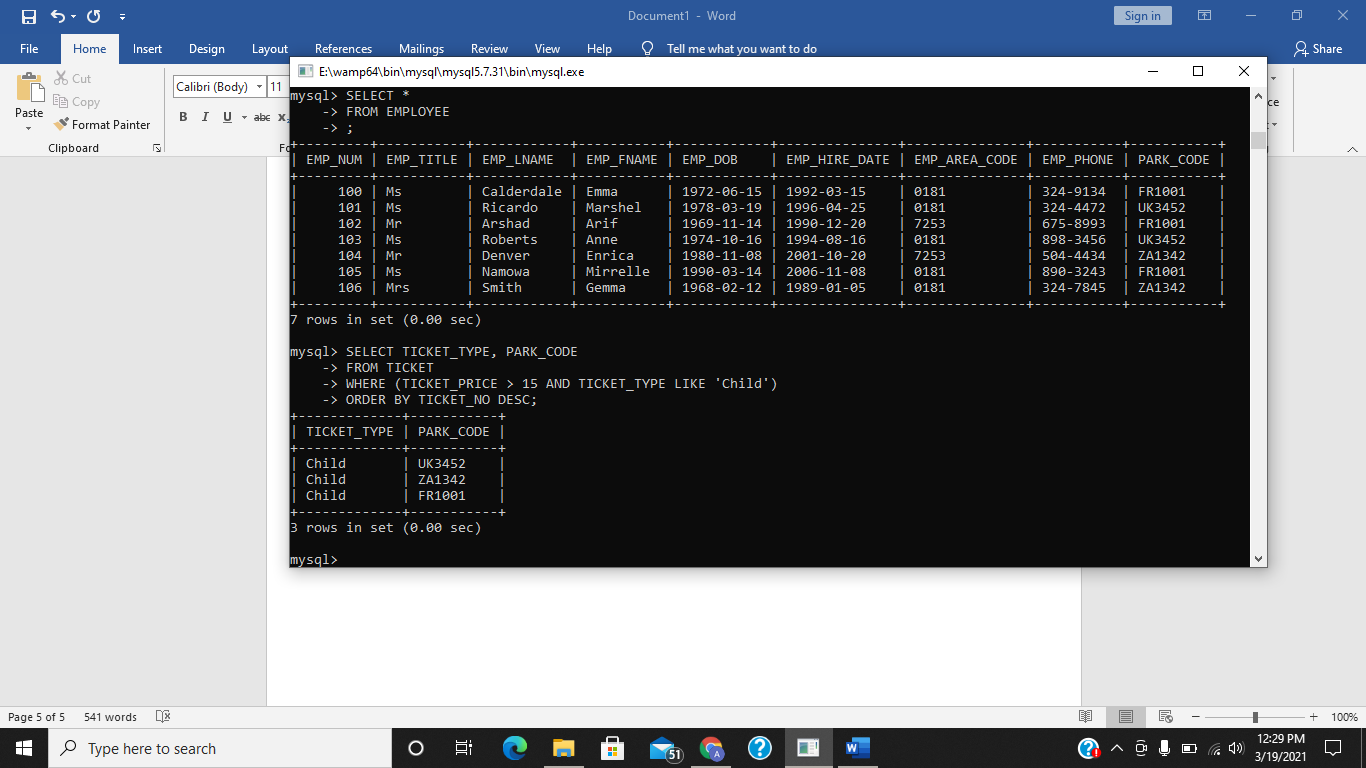
**Task 3.12 Enter the following query which contains an example of a cascading order sequence, by ordering the rows in the employee table by the employee’s last then first names.**

SELECT \* FROM EMPLOYEE ORDER BY EMP\_LNAME, EMP\_FNAME;



**Task 3.13 Enter the following query and check your output against the results shown in Figure 35.**

SELECT TICKET\_TYPE, PARK\_CODE FROM TICKET WHERE (TICKET\_PRICE > 15 AND TICKET\_TYPE LIKE 'Child') ORDER BY TICKET\_NO DESC;



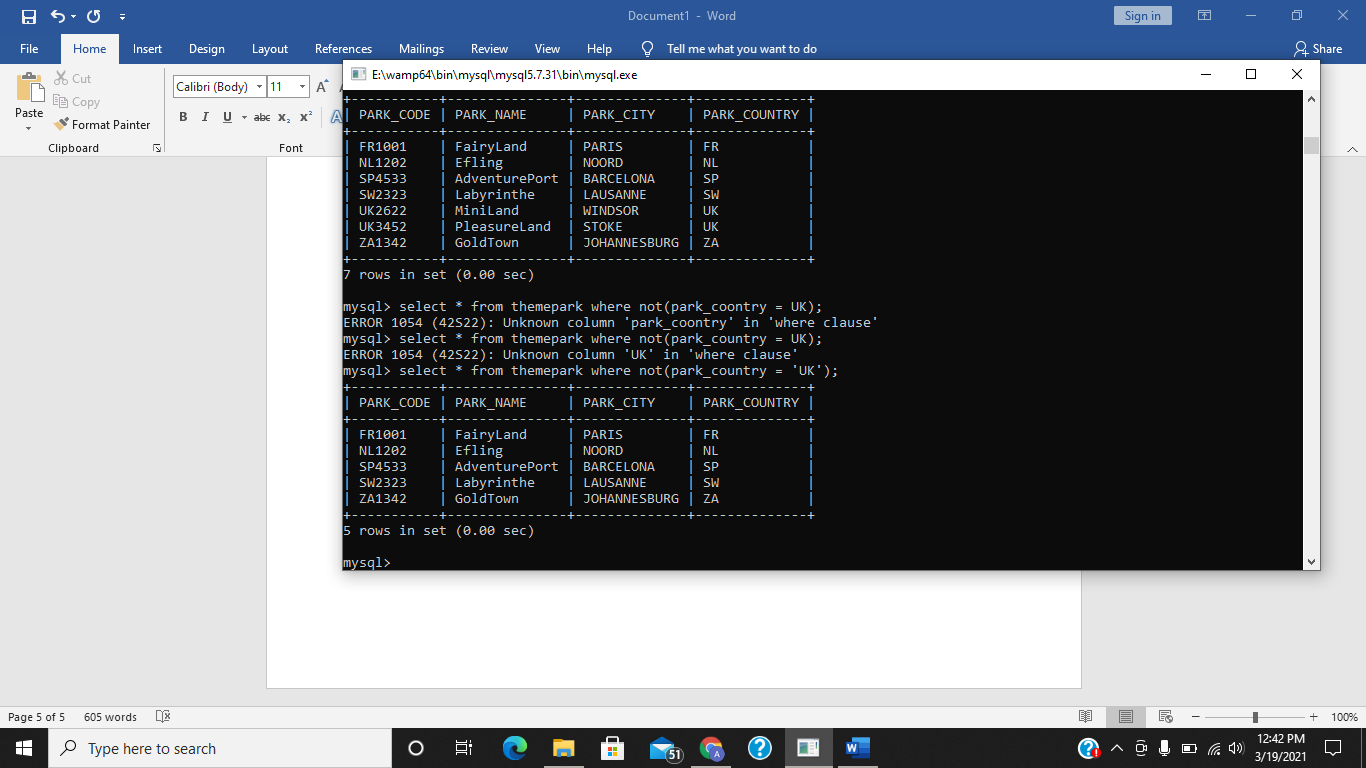
**Describe in your own words what this query is actually doing.**

This query is showing us data of columns TICKET\_TYPE, PARK\_CODE in which TICKET\_PRICE is greater then 15 and type of ticket is ‘Child’ and is sorting ticket in descending order.

**Exercises**

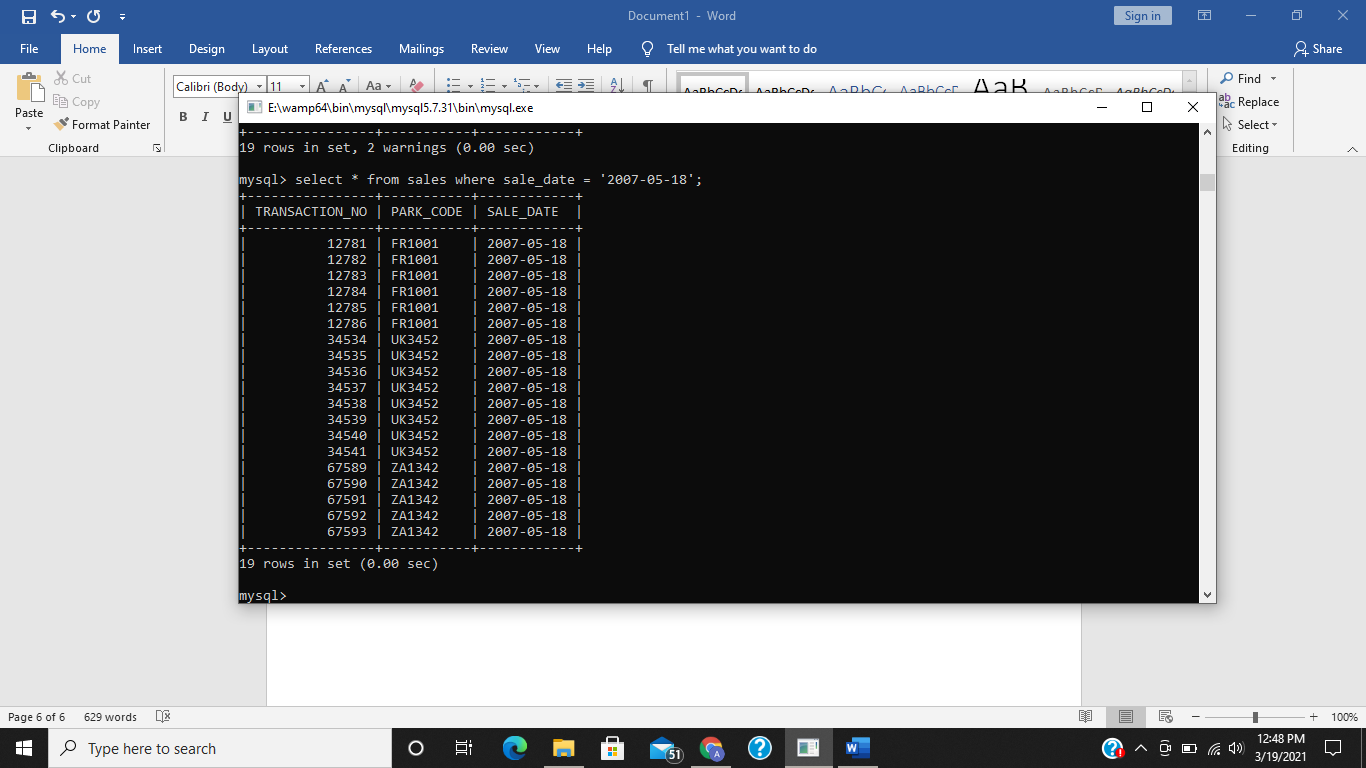
**E3.1 Write a query to display all Theme Parks except those in the UK**

select \* from themepark where not(park\_country = 'UK');



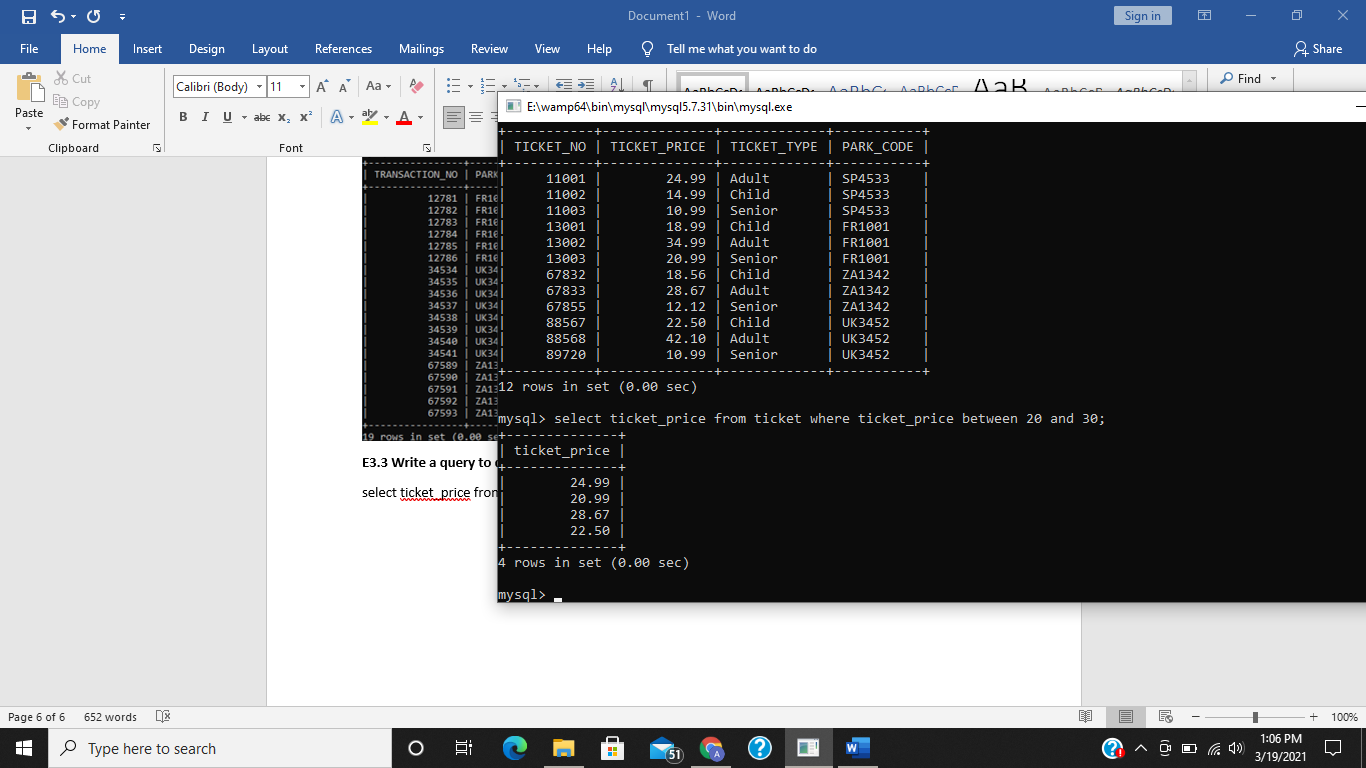
**E3.2 Write a query to display all the sales that occurred on the 18th May 2007.**

select \* from sales where sale\_date = '2007-05-18';



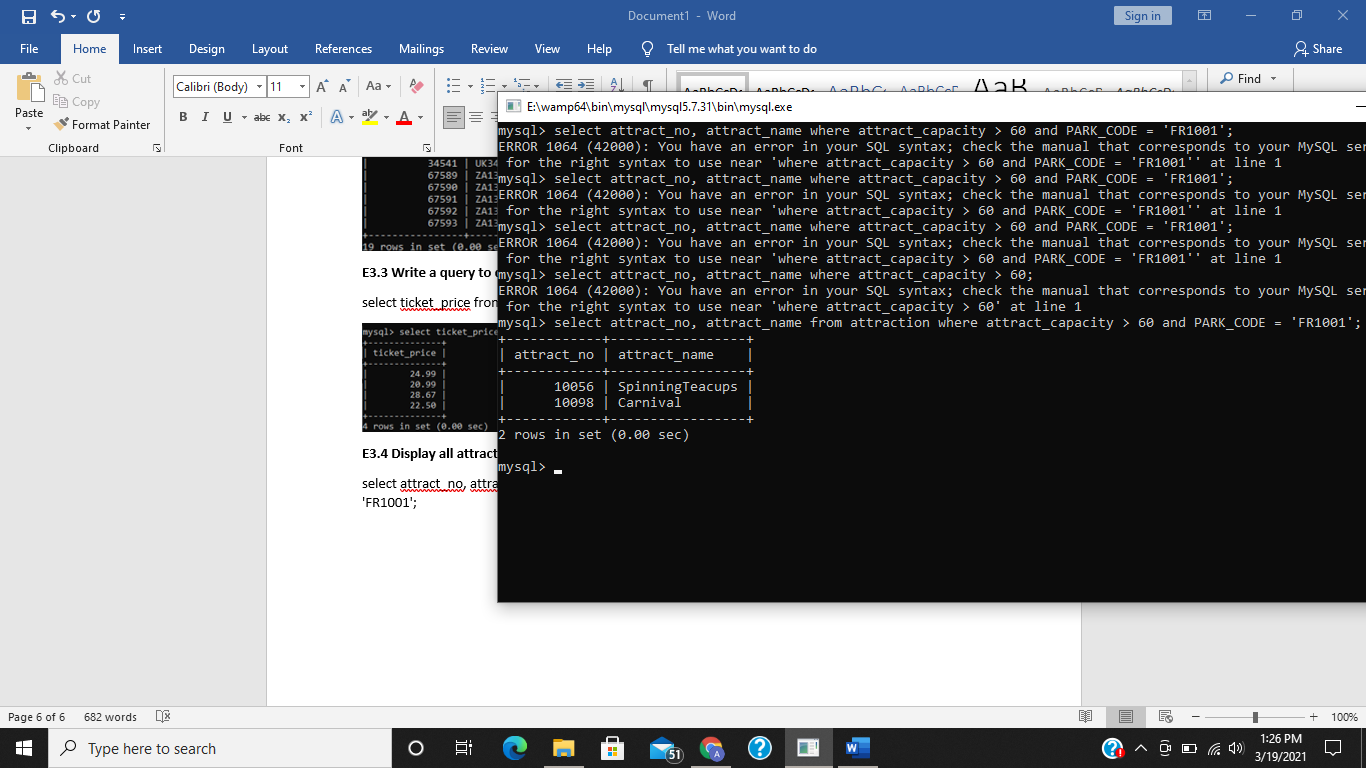
**E3.3 Write a query to display the ticket prices between €20 AND €30.**

select ticket\_price from ticket where ticket\_price between 20 and 30;



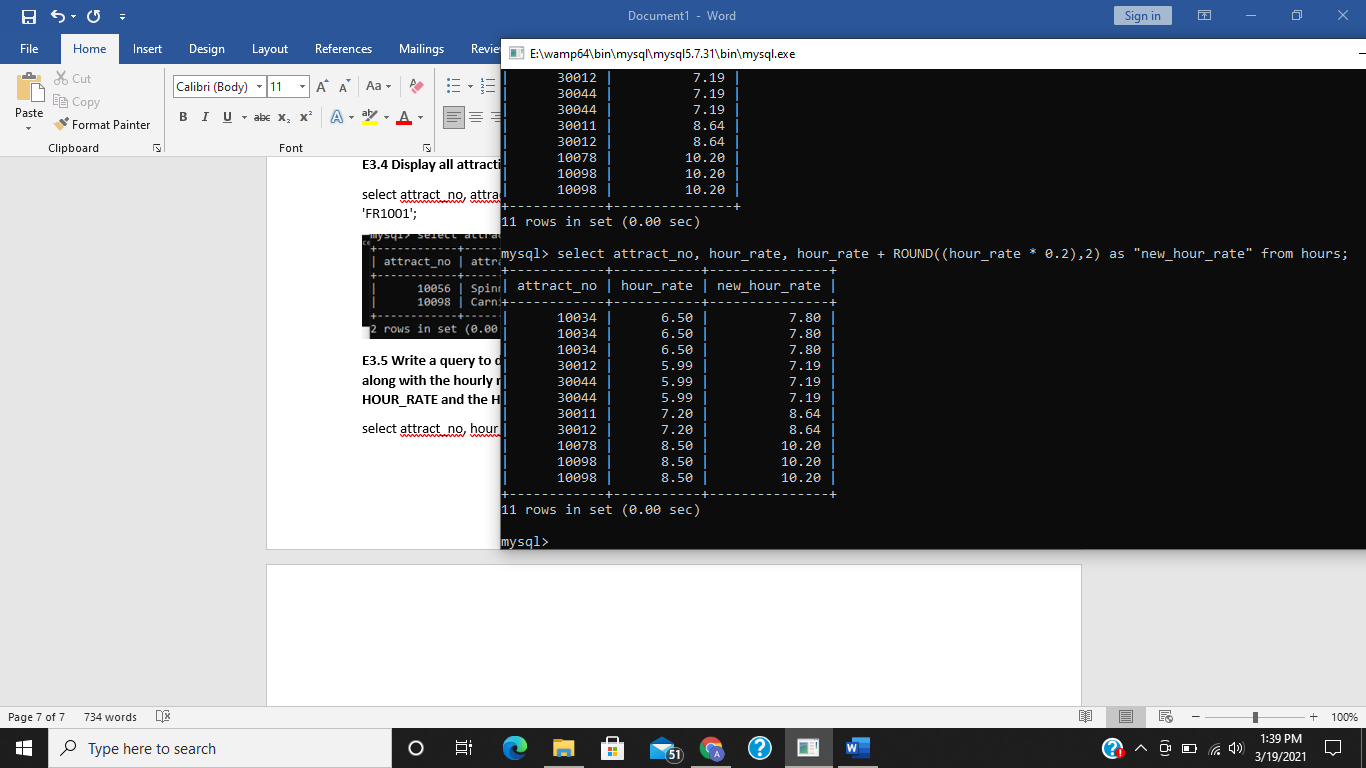
**E3.4 Display all attractions that have a capacity of more than 60 at the Theme Park FR1001**

select attract\_no, attract\_name from attraction where attract\_capacity > 60 and PARK\_CODE = 'FR1001';



**E3.5 Write a query to display the hourly rate for each attraction where an employee had worked, along with the hourly rate increased by 20%. Your query should only Display the ATTRACT\_NO, HOUR\_RATE and the HOUR\_RATE with the 20% Increase.**

select attract\_no, hour\_rate, hour\_rate + ROUND((hour\_rate \* 0.2),2) as "new\_hour\_rate" from hours;



**E.3.6 Elaborate Difference IN vs BETWEEN operators with examples.**

Differences between these operator is that the BETWEEN operator is used to select a range of data between two values while The IN operator allows you to specify multiple values.

**BETWEEN** **Operator**

The BETWEEN operator selects a range of data between two values. The values can be numbers, text,etc.

**Example**

SELECT \*FROM akash1

WHERE marks BETWEEN 50 AND 80

**IN** **Operator**

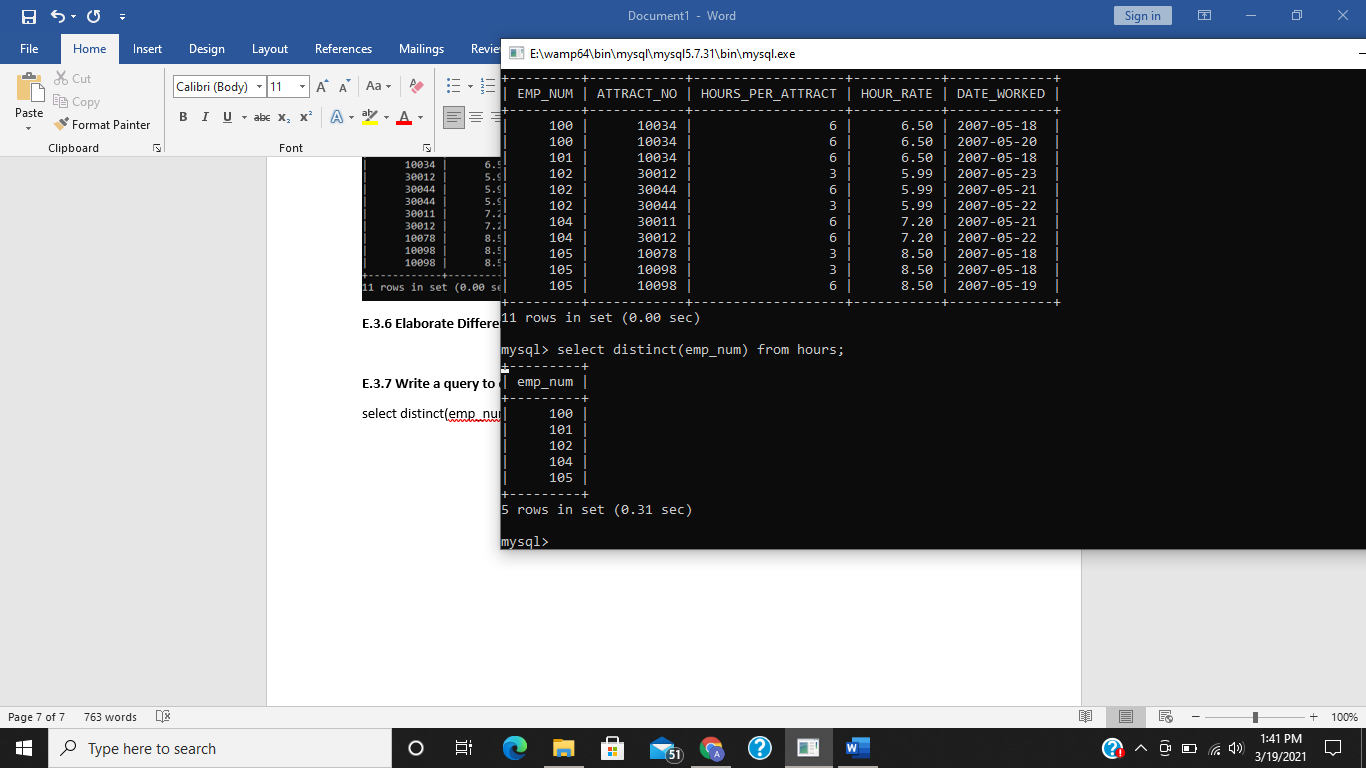
The IN operator allows you to specify multiple values.  
**Example**

SELECT \* FROM data

WHERE marks IN (89,81)

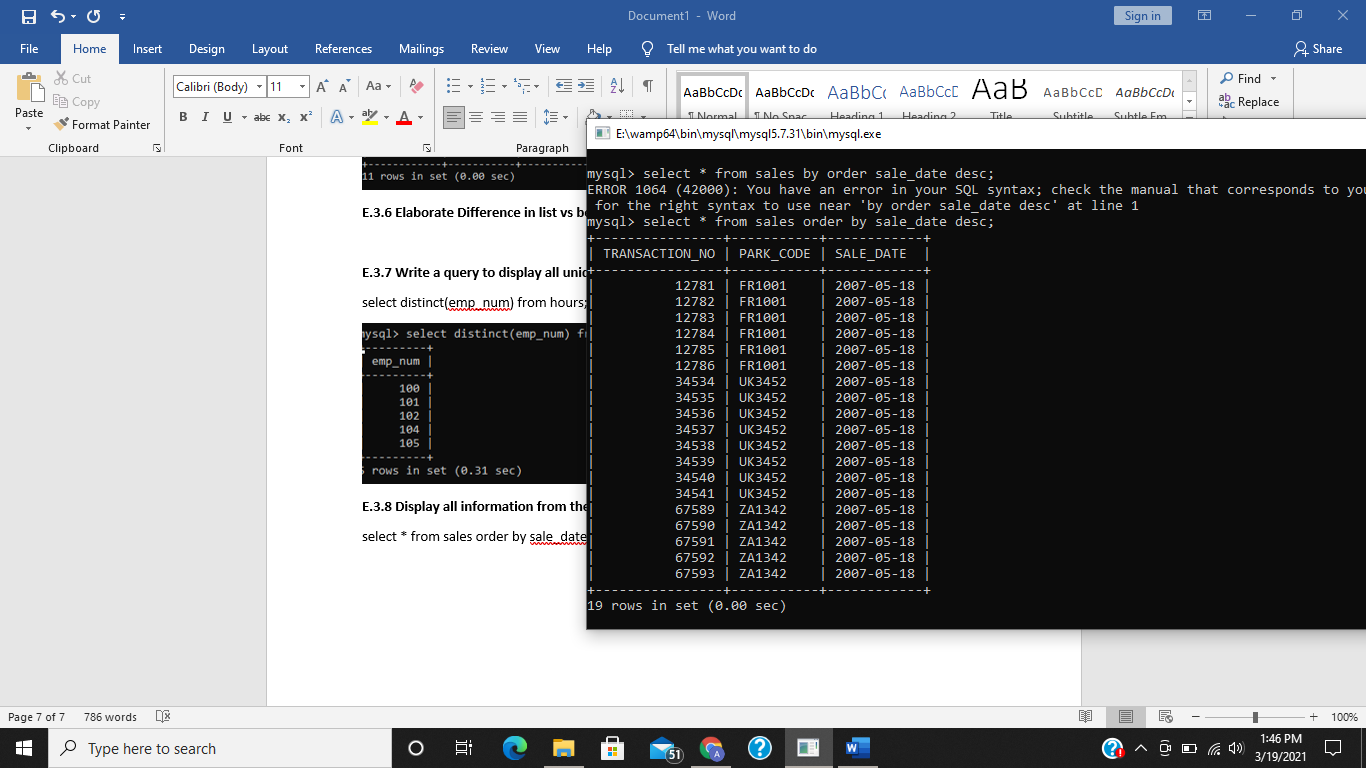
**E.3.7 Write a query to display all unique employees that exist in the HOURS table.**

select distinct(emp\_num) from hours;



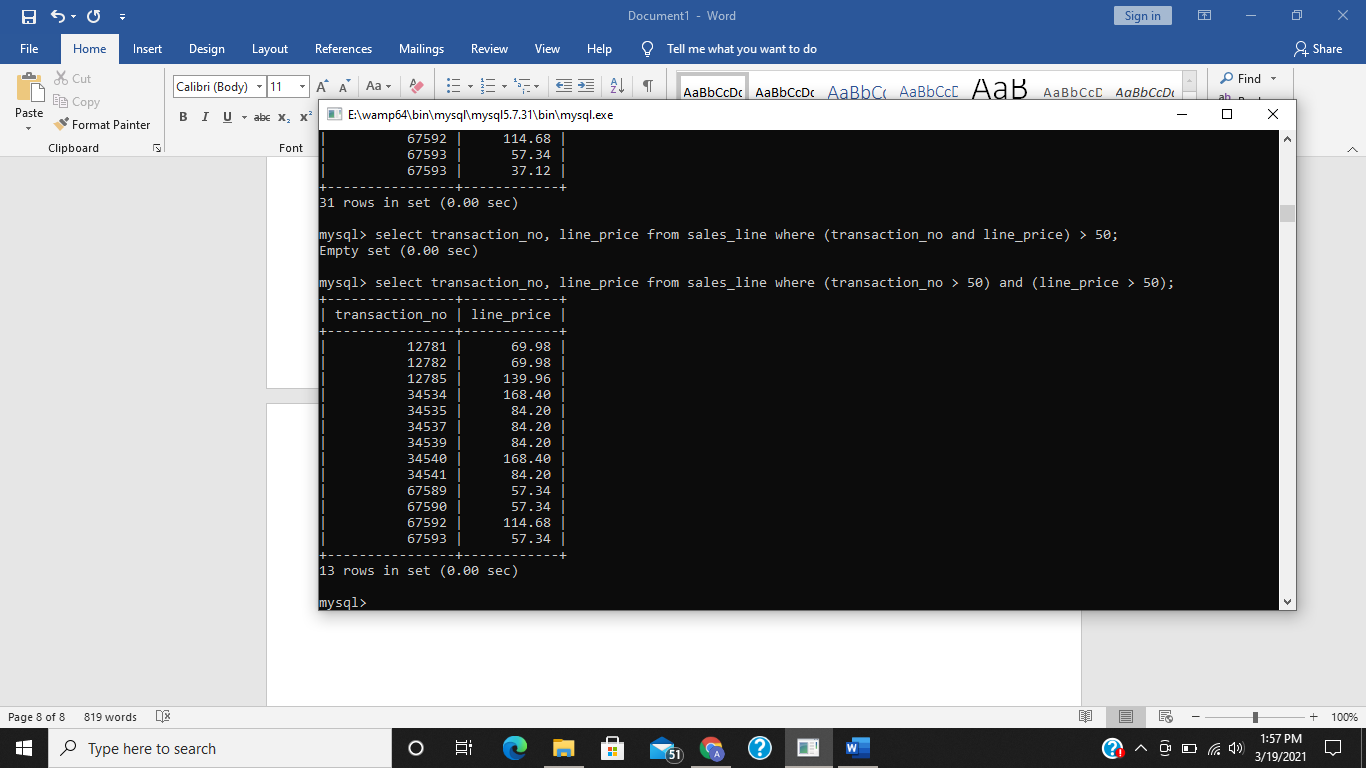
**E.3.8 Display all information from the SALES table in descending order of the sale date.**

select \* from sales order by sale\_date desc;



**E.3.9 Write a query to show the transaction numbers and lineprices (in the SALES\_LINE table) that are greater than €50.**

select transaction\_no, line\_price from sales\_line where (transaction\_no > 50) and (line\_price > 50);



**E.3.10 Write a query to display only the last two Employee Record (EMP\_NUM,EMP\_FNAME) in descending order.**

select emp\_num, emp\_fname from employee order by emp\_num desc, emp\_fname desc;

