NUR ALEEYA ALEESYA BINTI HEZRIE

2022892198

CDCS1103C

1. Write a query to calculate the number of unique park code that exist in the TICKET table, replace calculated column with 'Park_Count'.

```
SELECT COUNT(DISTINCT(PARK_CODE)) AS 'PARK_COUNT'
FROM TICKET;
```

Result Grid

	PARK_COUNT
▶	4

2. Display the employee numbers of all employees and the total payment (Hour rate multiply Hour per attract) they have worked. Use 'Total payment' for calculated column. Sort the output according employee number descending.

```
SELECT EMP_NUM, ROUND(HOUR_RATE * HOURS_PER_ATTRACT) AS 'TOTAL PAYMENT'
FROM HOURS
ORDER BY EMP_NUM DESC;
```

Result Grid | Filter Rows:

	EMP_NUM	TOTAL PAYMENT
▶	105	51
	105	26
	105	26
	104	43
	104	43
	102	36
	102	18
	101	39
	100	39

3. Display the employee numbers of all employees and the total hours they have worked. Use 'Total Hours' for calculated column. Sort the output according employee number descending.

```
SELECT EMP_NUM,
SUM(HOURS_PER_ATTRACT) AS 'TOTAL HOURS'
FROM HOURS
GROUP BY EMP_NUM
ORDER BY EMP_NUM DESC;
```

Result Grid | Filter Rows:

EMP_NUM	TOTAL HOURS
105	12
104	12
102	9
101	6
100	6

4. Show the attraction number and the minimum and maximum hourly rate for each attraction. Use alias MIN and MAX for calculated column.

```
SELECT ATTRACT_NO, MIN(HOUR_RATE) AS 'MIN',
MAX(HOUR_RATE) AS 'MAX'
FROM HOURS
GROUP BY ATTRACT_NO;
```

Result Grid | Filter Rows:

ATTRACT_NO	MIN	MAX
10034	6.50	6.50
10078	8.50	8.50
10098	8.50	8.50
30011	7.20	7.20
30012	5.99	8.50
30044	5.99	5.99

5. Write a query to show the transaction numbers and AVERAGE line prices (use the SALES_LINE table). Display average value that are greater than €50 only.

```
SELECT TRANSACTION_NO,
AVG(LINE_PRICE) AS 'AVERAGE'
FROM SALES_LINE
GROUP BY TRANSACTION_NO
HAVING AVG(LINE_PRICE) > 50;
```

Result Grid | Filter Rows:

TRANSACTION_NO	AVERAGE
12782	69.980000
12785	63.313333
34534	70.960000
34535	84.200000
34537	53.350000
34539	53.090000
34540	70.960000
34541	84.200000
67592	114.680000

6. Display Transaction Number, Sale date information from the SALES table. Calculate the number of sales after 1 January 2007. Sort the result in descending order of the sale date.

```
SELECT TRANSACTION_NO, SALE_DATE,
COUNT(*) AS 'NUMBER OF SALES'
FROM SALES
WHERE SALE_DATE > '2007-01-01'
GROUP BY TRANSACTION_NO, SALE_DATE
ORDER BY SALE_DATE DESC;
```

Result Grid | Filter Rows: | Export:

TRANSACTION_NO	SALE_DATE	NUMBER OF SALES
12781	2007-05-18	1
12782	2007-05-18	1
12783	2007-05-18	1
12784	2007-05-18	1
12785	2007-05-18	1
12786	2007-05-18	1
34534	2007-05-18	1
34535	2007-05-18	1
34536	2007-05-18	1
34537	2007-05-18	1
34538	2007-05-18	1
34539	2007-05-18	1
34540	2007-05-18	1
34541	2007-05-18	1
67589	2007-05-18	1
67590	2007-05-18	1
67591	2007-05-18	1
67592	2007-05-18	1
67593	2007-05-18	1

Result 52

7. Using the TICKET table, write a query to display the park code and the average ticket price. Limiting the average ticket price greater or equal to 20. Replace calculated column name using 'AVERAGE PRICE'.

```
SELECT PARK_CODE,
AVG(TICKET_PRICE) AS 'AVERAGE PRICE'
FROM TICKET
GROUP BY PARK_CODE
HAVING AVG(TICKET_PRICE) >= 20;
```

Result Grid | Filter Rows:

PARK_CODE	AVERAGE PRICE
FR1001	24.990000
UK3452	25.196667