1.Find marks of students within a range

You want to find the details of students based on their MARKS.

Write a SQL query to display details of all students who have scored MARKS within the range **400 and**
You want to find the details of students based on their MARKS.

Write a SQL query to display details of all students who have scored MARKS within the range 400 and 6000 except those whose MARKS are 1200 and 5236.

STUDENT_ID	INT	Column denoting STUDENT_ID representing id of the student
STUDENT_NAME	VARCHAR(50)	Column denoting STUDENT_NAME representing name of the student
MARKS	INT	Column denoting MARKS representing marks scored by the student
CITY	VARCHAR(50)	Column denoting CITY representing city in which student resides

```
-- Enter your query here
select  * from students where marks between 400 and 6000 and marks not in (1200,5
236);
```

2.Display the details of all products

You want to do a product price analysis based on the manufacturing date.
Write a SQL query to display the details of all products for which the **PROD_PRICE** is greater than at least one of the products manufactured on **31st July 2018**.

Product

Name	Type	Description
PRO_NO	int(10)	Column denoting PRO_NO representing product number
PROD_NAME	VARCHAR(50)	Column denoting PROD_NAME representing the name of the product
MANU_DATE	DATE	Column denoting MANU_DATE representing the date on which the product was manufactured
PROD_PRICE	int(10)	Column denoting PROD_PRICE representing the price of the product
SALES_ID	int(10)	Column denoting SALES_ID representing id of a salesman selling the product

```
SELECT * FROM PRODUCT WHERE PROD_PRICE > (SELECT MIN(PROD_PRICE) FROM PRODUCT WHE
RE MANU_DATE='2018-07-31');
```

3. Details of the most expensive products

You want to display product details based on manufacturing company.

Write a SQL query to display the **PROD_NAME**, **PROD_PRICE** and **COM_NAME** for the most expensive product of each company only if they have **COM_ID** which is present in both the given tables.

COMPANY

Name	Type	Description
COM_ID	int(10)	Column denoting COM_ID representing id of the company.
COM_NAME	VARCHAR(50)	Column denoting COM_NAME representing the name of the company.

PRODUCT

Name	Type	Description
PROD_ID	int(10)	Column denoting PROD_ID representing id of the product.
PROD_NAME	VARCHAR(50)	Column denoting PROD_NAME representing name of the product.
PROD_PRICE	int(10)	Column denoting PROD_PRICE representing price of the product.
COM_ID	int(10)	Column denoting COM_ID representing the id of the company.

```
SELECT  PROD_NAME,PROD_PRICE,COM_NAME FROM PRODUCT JOIN COMPANY ON PRODUCT.COM_ID
=COMPANY.COM_ID WHERE PROD_PRICE=(SELECT MAX(PROD_PRICE)FROM PRODUCT P1 WHERE P1.
COM_ID=PRODUCT.COM_ID)
```

4. Comparative analysis of employee salaries

You want to do a comparative analysis of employee salaries across all departments.

Write a SQL query to display the **EMP_NAME**, **EMP_SALARY** and **DEPT_ID** of those employees whose EMP_SALARY is greater than or equal to the EMP_SALARY of the employee whose EMP_NO is **equal to 103**.

```
CREATE TABLE EMPLOYEE(
  EMP_NO int(10) PRIMARY KEY,
  EMP_NAME VARCHAR(50),
  HIRE_DATE DATE,
  EMP_SALARY int(10),
  DEPT_ID int(10)
)
```

EMPLOYEE

Name	Type	Description
EMP_NO	int(10)	Column denoting EMP_NO representing Employee number
EMP_NAME	VARCHAR(50)	Column denoting EMP_NAME representing name of employee
HIRE_DATE	DATE	Column denoting HIRE_DATE representing date on which employee is hired
EMP_SALARY	int(10)	Column denoting EMP_SALARY representing salary of the employee
DEPT_ID	int(10)	Column denoting DEPT_ID representing id of the department where the employee works

```
SELECT EMP_NAME,EMP_SALARY,DEPT_ID FROM EMPLOYEE  WHERE EMP_SALARY >=(SELECT EMP_
SALARY FROM EMPLOYEE WHERE EMP_NO=103)
```

5. Select based on Cases and Date function

You want to filter product details based on modified PROD_PRICE.

Write a MySQL query to display PRO_NO, PROD_NAME, MANU_DATE and modified PROD_PRICE of the products whose modified PROD_PRICE lies between 1000 and 7000.

Note: The modified PROD_PRICE is calculated as follows:

1. If manufacturing day is Wednesday, Modified PROD_PRICE is PROD_PRICE increased by 100.
2. If manufacturing day is Sunday, Modified PROD_PRICE is PROD_PRICE increased by 50.
3. If manufacturing day is Saturday, Modified PROD_PRICE is PROD_PRICE increased by 700.
4. Else Modified PROD_PRICE is PROD_PRICE increased by 500.

Schema

```
PRODUCT,
CREATE TEMPORARY TABLE ` PRODUCT ` (
  ` PRO_NO ` int NOT NULL,
  ` PROD_NAME ` varchar(50) DEFAULT NULL,
  ` MANU_DATE ` date DEFAULT NULL,
  ` PROD_PRICE ` int DEFAULT NULL,
  ` SALES_ID ` int DEFAULT NULL,
  PRIMARY KEY (` PRO_NO `)
) ENGINE = InnoDB DEFAULT CHARSET = latin1
```

PRODUCT

Name	Type	Description
PRO_NO	int	Column denoting PRO_NO representing product number
PROD_NAME	varchar(50)	Column denoting PROD_NAME representing product name
MANU_DATE	date	Column denoting MANU_DATE representing manufacturing date

Name	Type	Description
PROD_PRICE	int	Column denoting PROD_PRICE representing price of product
SALES_ID	int	Column denoting SALES_ID representing is of salesman selling the product

```
SELECT PRO_NO ,PROD_NAME,MANU_DATE,
CASE
WHEN DAYOFWEEK(MANU_DATE)=4 THEN PROD_PRICE +100
WHEN DAYOFWEEK(MANU_DATE)=1 THEN PROD_PRICE +50
WHEN DAYOFWEEK(MANU_DATE)=7 THEN PROD_PRICE +700
ELSE PROD_PRICE +500
END AS MODIFIED_PROD_PRICE
FROM PRODUCT
WHERE
(CASE
WHEN DAYOFWEEK(MANU_DATE)=4 THEN PROD_PRICE +100
WHEN DAYOFWEEK(MANU_DATE)=1 THEN PROD_PRICE +50
WHEN DAYOFWEEK(MANU_DATE)=7 THEN PROD_PRICE +700
ELSE PROD_PRICE +500
END)
BETWEEN 1000 AND 7000;
```

6. Finding total number of employees

You want to find the total number of employees based on the hiring date.
Write a MySQL query to display the DEPT_NAME and the total number of employees of the departments that follow the given condition.

Condition: *The total number of employees in the department must be more than the total number of employees who are hired on 2020-11-11.*

Schema

```
CREATE TABLE DEPARTMENT(DEPT_ID int(10), DEPT_NAME VARCHAR(50))

CREATE TABLE EMPLOYEE(
  EMP_NO int(10) PRIMARY KEY,
  EMP_NAME VARCHAR(50),
  HIRE_DATE DATE,
  EMP_SALARY int(10),
  DEPT_ID int(10)
)
```

Table structure
DEPARTMENT

Name	Type	Description
DEPT_ID	int(10)	Column denoting DEPT_ID representing id of the department
DEPT_NAME	VARCHAR(50)	Column denoting DEPT_NAME representing name of the department

EMPLOYEE

Name	Type	Description
EMP_NO	int(10)	Column denoting EMP_NO representing employee number

Name	Type	Description
EMP_NAME	VARCHAR(50)	Column denoting EMP_NAME representing name of employee
HIRE_DATE	DATE	Column denoting HIRE_DATE representing date on which employee is hired
EMP_SALARY	int(10)	Column denoting EMP_SALARY representing salary of the employee
DEPT_ID	int(10)	Column denoting DEPT_ID representing id of the department where the employee works