**CPSC 4800**

**Final Exam**

**Total Time allowed:**

**1 hour and 45 minutes**

**Student Name: _______PHAN HOANG AN NGUYEN______________________**

**Student ID: ______________________100404103_________________________**

**Instructions:**

1. Questions 1 through 7 are related to Database/SQL, the rest of the questions are related to Python.
2. You should answer and submit the SQL questions in the word document.
3. You should answer and submit the Python questions in the Jupyter notebook file.
4. The exam will be open book. You can look at all the course materials, labs, solutions, etc on D2L. However, you are not allowed to communicate or receive help from anyone else during the exam.
5. The work submitted must be your own. Plagiarism and cheating will be dealt with very seriously.
6. If the instructor realizes that a student cheated during the exam, he/she will receive a grade of zero for the final.

**Warning:**
**By submitting an exam file, you promise that the code you have submitted is your own and you did not communicate with or receive help from anyone else.**

| Question | Points | Score |
|---|---|---|
| **SQL** | | |
| **1** | 20 | |
| **2** | 2 | |
| **3** | 2 | |
| **4** | 6 | |
| **5** | 10 | |
| **6** | 10 | |
| **7** | 10 | |
| **Python** | | |
| **8** | 10 | |
| **9** | 10 | |
| **10** | 10 | |
| **11** | 10 | |

**Question 1 –** The dataset is related to this Kaggle link. https://www.kaggle.com/uciml/restaurant-data-with-consumer-ratings?select=rating_final.csv

Download final_rating.csv and userprofile.csv from the final exam folder. You need to write the SQL codes to answer the below questions. You can create the schemas and import the datasets to pgadmin to check your SQL codes or you could just write them here without creating the tables, we don't grade the table creations for thus question, we just grade your SQL codes.

a) Find the average height of the users whose rating was 2 (round the average numbers to 2 decimal places).

```
SELECT userprofile.userID,

        round(avg(userprofile.height), 2) AS AvgHeightRating2

FROM userprofile

JOIN final_rating ON userprofile.userID = final_rating.userID

WHERE final_rating.rating = 2
GROUP BY userprofile.userID;
```

b) How many times each user has rated? Sort out the values in the descending order.

```
SELECT userID, count(rating)

FROM final_rating

GROUP BY userID;
```

c) Compare the average food rating, and service rating of the users with various budget levels.

```
SELECT userprofile.budget,

     AVG(final_rating.food_rating),

     AVG(final_rating.service_rating)

FROM userprofile

JOIN final_rating ON userprofile.userID = final_rating.userID

GROUP BY userprofile.budget;
```

d) Find the average rating of the users who have been smoking?

```
SELECT userprofile.smoker,

     AVG(final_rating.rating),
```

FROM userprofile

JOIN final_rating ON userprofile.userID = final_rating.userID

GROUP BY userprofile.smoker

HAVING userprofile.smoker = ‘TRUE’;

Consider the Tables below and write the SQL queries. Answer each question and paste the answer as asked in the blanks.

**TABLE: Employee**

| employee_id | first_name | last_name | salary | joining_date | department |
|---|---|---|---|---|---|
| 1 | Jack | Zu | 75000 | 2013-01-01 | MANAGEMENT |
| 2 | Ashley | Leach | 90000 | 2013-01-01 | SERVICES |
| 3 | Amna | Robles | 95000 | 2013-02-01 | SERVICES |
| 4 | Martin | Mullen | 65000 | 2014-02-01 | BANKING |
| 5 | Jerry | Pinto | 70000 | 2015-03-01 | BANKING |
| 6 | Alex | Baran | 80000 | 2014-02-01 | MANAGEMENT |
| 7 | John | Smith | 100000 | 2013-01-01 | DIRECTOR |

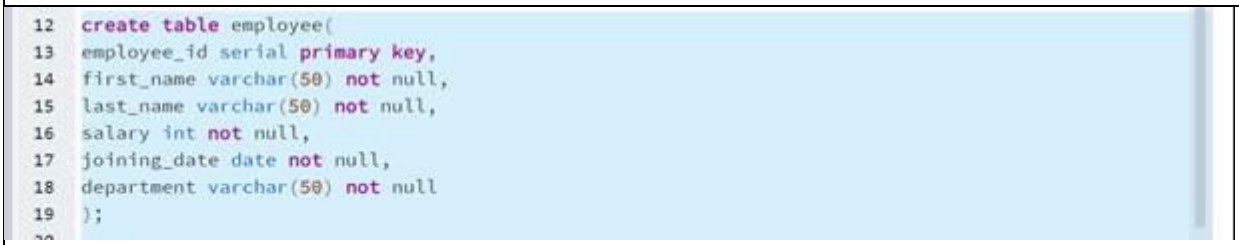
**TABLE: Incentive**

| emp_ref_id | incentive_date | incentive_amount |
|---|---|---|
| 7 | 2014-02-01 | 10000 |
| 2 | 2014-02-01 | 5000 |
| 3 | 2014-02-01 | 4500 |
| 1 | 2015-03-01 | 4000 |
| 2 | 2015-03-01 | 3500 |

**Question 2 -** Create Tables employee and incentive in the database ‘employee_records’.

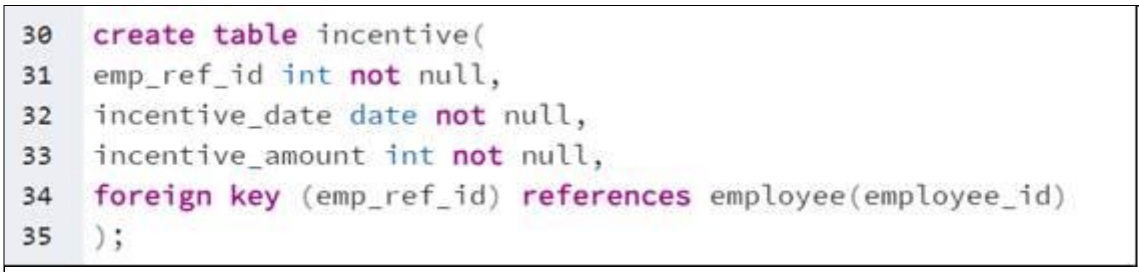
Make sure to define the respective PRIMARY and FOREIGN keys wherever applicable.

Paste create table Query in text format for Table employee



```
create table employee(
employee_id serial primary key,
first_name varchar(50) not null,
last_name varchar(50) not null,
salary int not null,
joining_date date not null,
department varchar(50) not null
);
```

Paste create table Query in text format for Table incentives



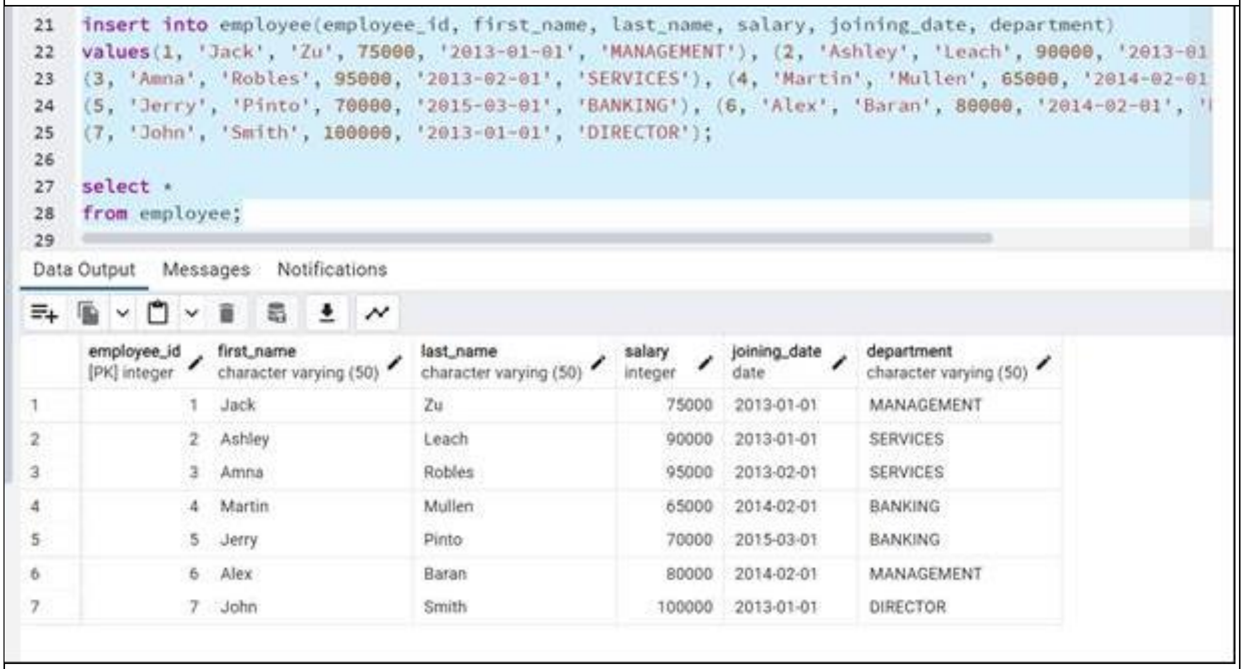
```
create table incentive(
emp_ref_id int not null,
incentive_date date not null,
incentive_amount int not null,
foreign key (emp_ref_id) references employee(employee_id)
);
```

**Question 3 -** To insert data into the employee table, you have two options:

1- Download the employee.csv file and import the data to the table OR

2- Manually insert the values into the table using **PostgreSQL INSERT**

After importing the data successfully into employee, or inserting the values, run SELECT * FROM employee and paste the resulting screenshot here.



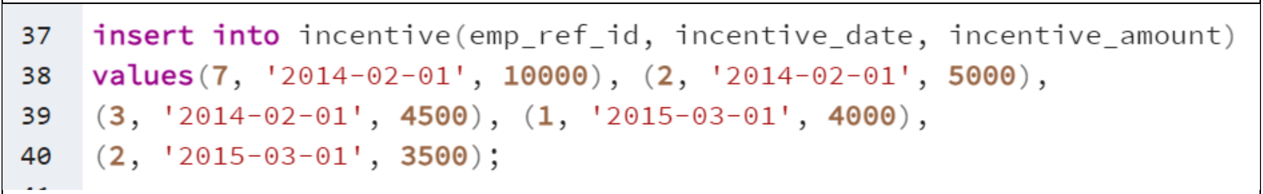
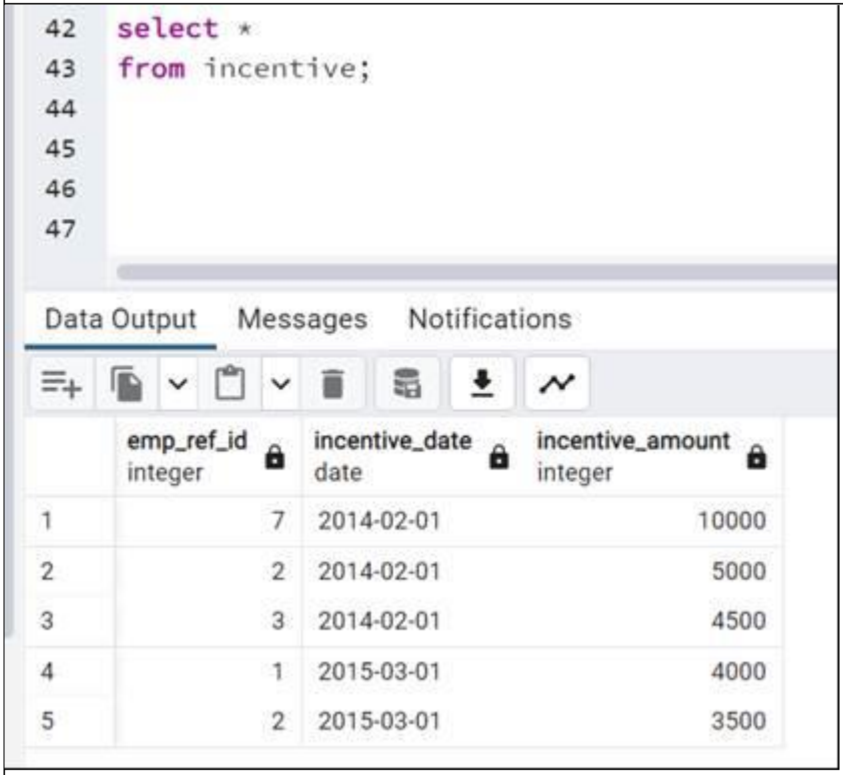
```
insert into employee(employee_id, first_name, last_name, salary, joining_date, department)
values(1, 'Jack', 'Zu', 75000, '2013-01-01', 'MANAGEMENT'), (2, 'Ashley', 'Leach', 90000, '2013-01
(3, 'Amna', 'Robles', 95000, '2013-02-01', 'SERVICES'), (4, 'Martin', 'Mullen', 65000, '2014-02-01
(5, 'Jerry', 'Pinto', 70000, '2015-03-01', 'BANKING'), (6, 'Alex', 'Baran', 80000, '2014-02-01', '

select *
from employee;
```

Data Output | Messages | Notifications

| | employee_id [PK] integer | first_name character varying (50) | last_name character varying (50) | salary integer | joining_date date | department character varying (50) |
|---|---|---|---|---|---|---|
| 1 | 1 | Jack | Zu | 75000 | 2013-01-01 | MANAGEMENT |
| 2 | 2 | Ashley | Leach | 90000 | 2013-01-01 | SERVICES |
| 3 | 3 | Amna | Robles | 95000 | 2013-02-01 | SERVICES |
| 4 | 4 | Martin | Mullen | 65000 | 2014-02-01 | BANKING |
| 5 | 5 | Jerry | Pinto | 70000 | 2015-03-01 | BANKING |
| 6 | 6 | Alex | Baran | 80000 | 2014-02-01 | MANAGEMENT |
| 7 | 7 | John | Smith | 100000 | 2013-01-01 | DIRECTOR |

**Question 4 -** Use the **PostgreSQL INSERT** statement to insert new rows into incentive table (same data as specified in the associated table).

| Paste your insert query here. |
|---|
|  |
| Run SELECT * FROM incentive and paste the resulting screenshot here. |
|  |

```
insert into incentive(emp_ref_id, incentive_date, incentive_amount)
values(7, '2014-02-01', 10000), (2, '2014-02-01', 5000),
(3, '2014-02-01', 4500), (1, '2015-03-01', 4000),
(2, '2015-03-01', 3500);
```

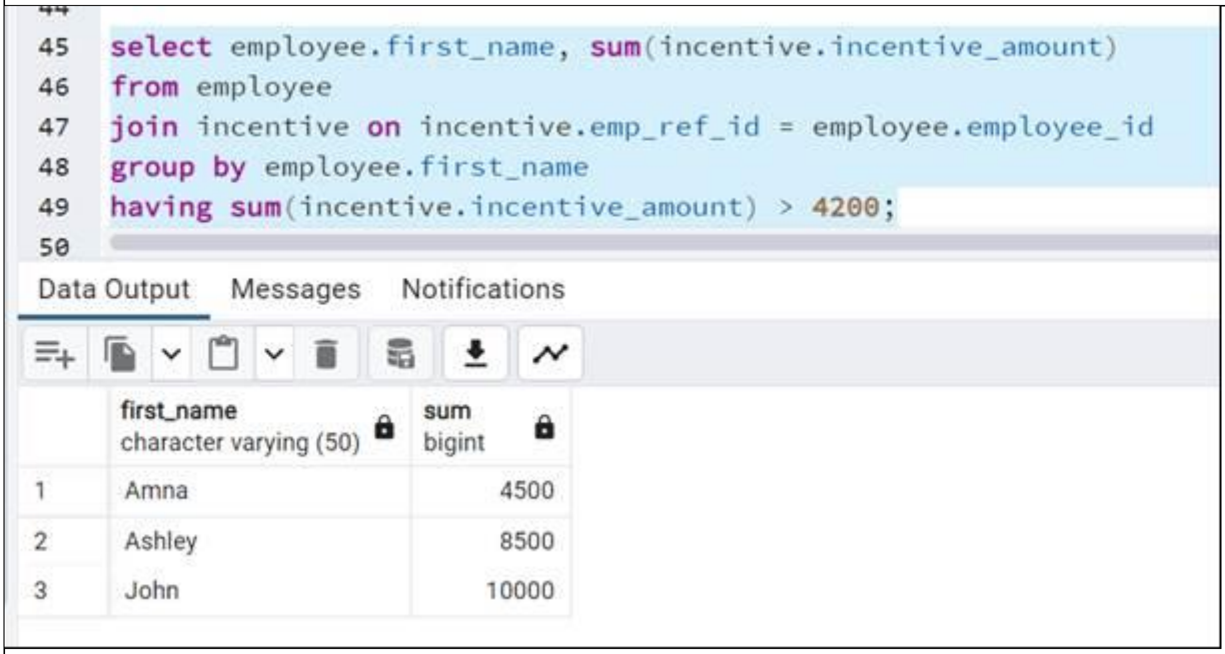
```
select *
from incentive;
```

| | emp_ref_id integer | incentive_date date | incentive_amount integer |
|---|---|---|---|
| 1 | 7 | 2014-02-01 | 10000 |
| 2 | 2 | 2014-02-01 | 5000 |
| 3 | 3 | 2014-02-01 | 4500 |
| 4 | 1 | 2015-03-01 | 4000 |
| 5 | 2 | 2015-03-01 | 3500 |

|  |
|---|
|  |

**Question 5 -** Select first_name, and total incentive amount for those employees who have total incentive amounts of greater than 4200.

Paste the screenshot of the query with the records in the below table.



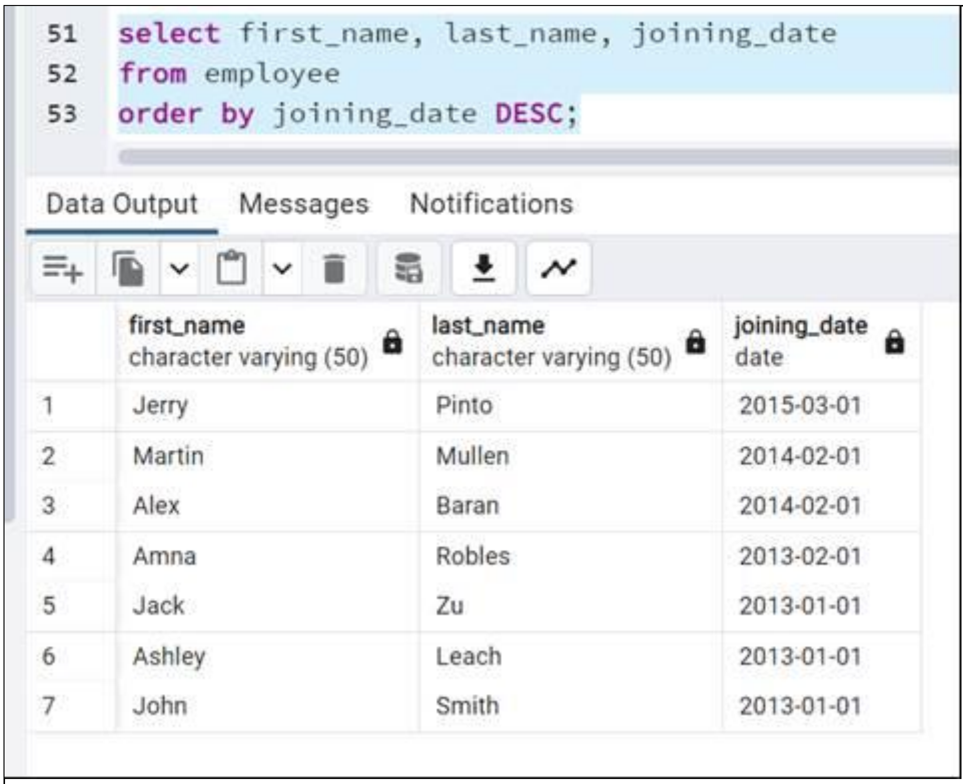
```
select employee.first_name, sum(incentive.incentive_amount)
from employee
join incentive on incentive.emp_ref_id = employee.employee_id
group by employee.first_name
having sum(incentive.incentive_amount) > 4200;
```

Data Output  Messages  Notifications

|  | first_name character varying (50) | sum bigint |
|---|---|---|
| 1 | Amna | 4500 |
| 2 | Ashley | 8500 |
| 3 | John | 10000 |

**Question 6-** Write SQL query to find and display the details of the employee who is junior in terms of the hiring date.

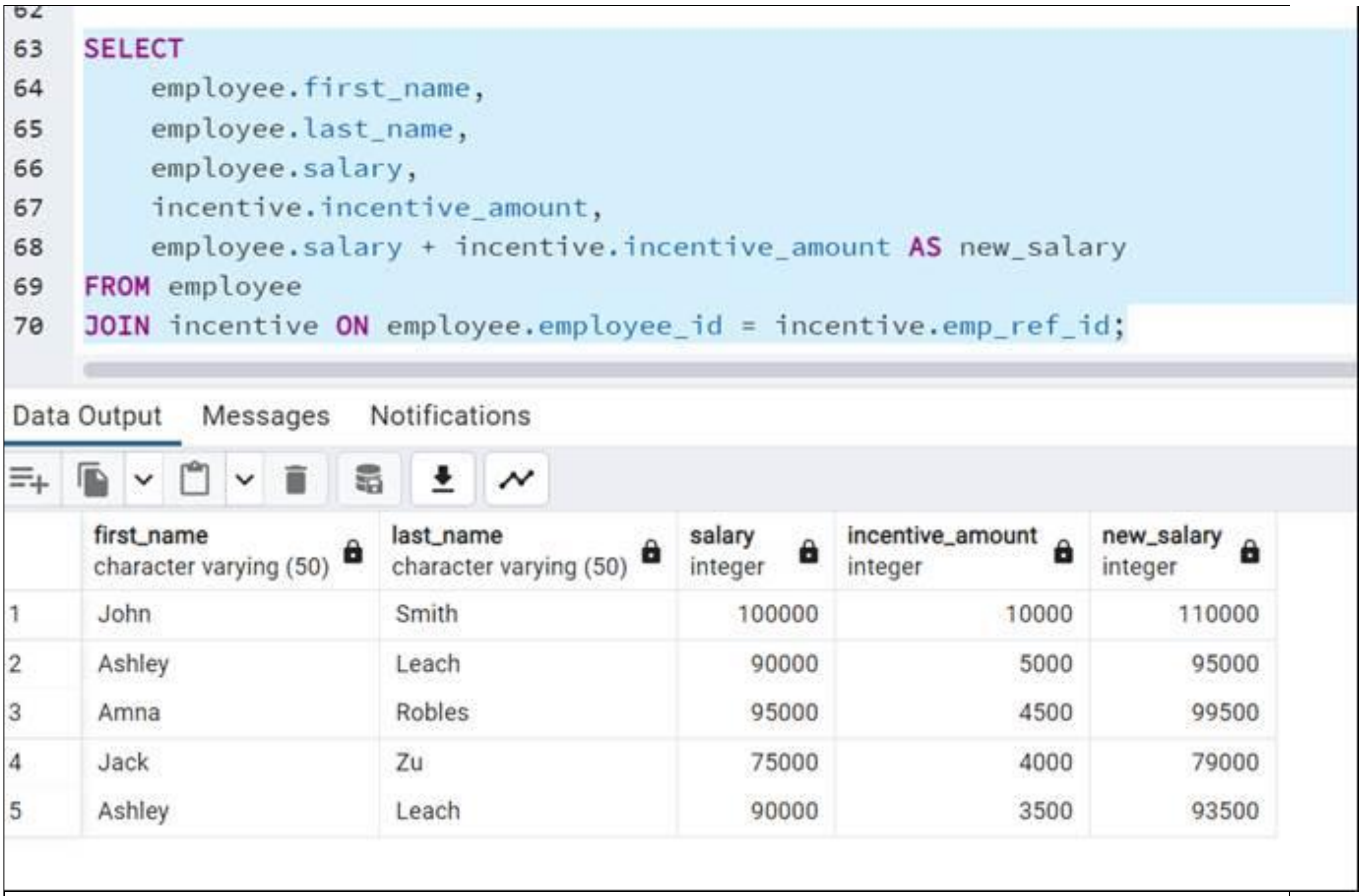
Paste the screenshot of the query with the records in the below table. → Tom Pinto is junior


```
select first_name, last_name, joining_date
from employee
order by joining_date DESC;
```

| | first_name character varying (50) | last_name character varying (50) | joining_date date |
|---|---|---|---|
| 1 | Jerry | Pinto | 2015-03-01 |
| 2 | Martin | Mullen | 2014-02-01 |
| 3 | Alex | Baran | 2014-02-01 |
| 4 | Amna | Robles | 2013-02-01 |
| 5 | Jack | Zu | 2013-01-01 |
| 6 | Ashley | Leach | 2013-01-01 |
| 7 | John | Smith | 2013-01-01 |

**Question 7**- Write a query to display the first_name, last_name, salary, incentive_amount and new salary (sum of incentive_amount and salary) for employees who have incentives.

| Paste the screenshot of the query with the records in the below table. |
|---|



```sql
SELECT
    employee.first_name,
    employee.last_name,
    employee.salary,
    incentive.incentive_amount,
    employee.salary + incentive.incentive_amount AS new_salary
FROM employee
JOIN incentive ON employee.employee_id = incentive.emp_ref_id;
```

Data Output Messages Notifications

| | first_name character varying (50) | last_name character varying (50) | salary integer | incentive_amount integer | new_salary integer |
|---|---|---|---|---|---|
| 1 | John | Smith | 100000 | 10000 | 110000 |
| 2 | Ashley | Leach | 90000 | 5000 | 95000 |
| 3 | Amna | Robles | 95000 | 4500 | 99500 |
| 4 | Jack | Zu | 75000 | 4000 | 79000 |
| 5 | Ashley | Leach | 90000 | 3500 | 93500 |