Problem 1

Sheet-6

(a) Query: All Employees per city.

This query requires a join between Store and Employee relation. Employee and Store table is in BCNF since, for Employee table, Name, Salary and SID depends on EID which is the superkey. For Store table, City and EID depends on SID. For the above query, we can simply put city column to Employee table to make denormalization. The denormalized schema would be:

⇒ Employee [EID, Name, Salary, SID, City]

⇒ We denormalized the table to avoid join operation for performance gain. Downside of this denormalization would be, we have redundant data for list of employees for city

⇒ To keep data consistent, for every store input, we need to enter corresponding store data in employee table also with city information.

⇒ Normal form of original schema is BCNF

⇒ Normal form of new schema is 2NF

⇒ Query after denormalization:

```
    Select * from employees
    orderby city
```

Query: Price of all item in given invoice IID = 5

This query requires a join between Item and Article table. Following decisions are made:-

⇒ To denormalize the table Item and Article to find the Price, we can simply put Price in Item table and the denormalized schema will be

ITEM(IID, SID, AID, Count, Price)

⇒ Downside of this denormalization would be, we need to insert redundant data to keep both

Article and Item consistent.

⇒ Query after denormalization would be

```
Select Price from ITEM where
  IID=5
```

⇒ Nomal form of original table is BCNF

⇒ Normal form of new table is 1NF, since, we inserted Price, which depends on AID, and AID is a partial key, so there exists partial FD.

(b)

(i) We can remove DISTINCT here since select * will essentially retrieve all rows.

(ii) As we are retrieving data only for EID 2002, we can simply omit GROUPBY and HAVING clause and put it in a WHERE clause. Also, when we include WHERE, there is no need for MAX aggregate also.

(iii) We can simply remove the join column and query over only Employee table to find the SID