Part1

Task 1:

Create the above database schema using CREATE TABLE statements, including primary key constraints, and the constraint that salary is integer in the range [5000,20000]. You can assume CHAR (20) type for all other attributes.

```
CREATE TABLE Employee (
    eid CHAR(20) PRIMARY KEY,
    name CHAR(20),
    salary INT CHECK (salary >= 5000 AND salary <= 20000),
    dept CHAR(20)
);

CREATE TABLE Sales (
    dept CHAR(20),
    item CHAR(20),
    PRIMARY KEY (dept, item)
);

CREATE TABLE Types (
    item CHAR(20),
    color CHAR(20),
    PRIMARY KEY (item, color)
);
```

Task2:

Insert the above records into the tables using INSERT statements.

```
INSERT INTO Employee (eid, name, salary, dept) VALUES
('111', 'Jane', 8000, 'Household'),
('222', 'Anderson', 8000, 'Toy'),
('333', 'Morgan', 10000, 'Cosmetics'),
('444', 'Lewis', 12000, 'Stationery'),
('555', 'Nelson', 6000, 'Toy'),
('666', 'Hoffman', 16000, 'Cosmetics');

INSERT INTO Sales (dept, item) VALUES
('Stationery', 'pen'),
('Cosmetics', 'lipstick'),
('Toy', 'puzzle'),
('Stationery', 'ink'),
('Household', 'disk'),
('Sports', 'skates'),
('Toy', 'lipstick');

INSERT INTO Types (item, color) VALUES
```

```sql
('pen', 'red'),
('lipstick', 'red'),
('pen', 'black'),
('puzzle', 'black'),
('ink', 'red'),
('ink', 'blue');
```

Task3:

1. Compute the maximum salary for each department that sells at least two distinct items.

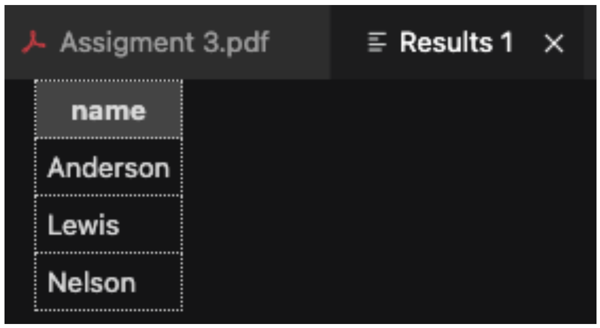
```sql
-- 1.3.1
SELECT dept, MAX(salary)
FROM Employee
WHERE dept IN (
        SELECT dept
        FROM Sales
        GROUP BY dept
        HAVING COUNT(DISTINCT item) >= 2
    )
GROUP BY dept;
```

Assigment 3.pdf Results

dept	MAX(salary)
Toy	8000
Stationery	12000

2. Compute the names of the employees who work in a department that sells some item in black color

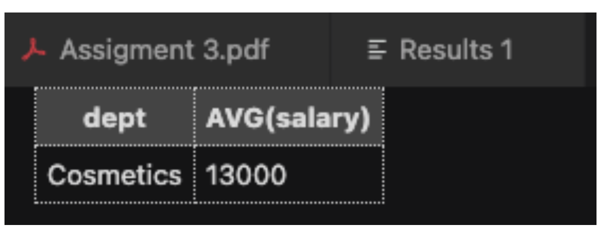
```sql
-- 1.3.2
SELECT DISTINCT E.name
FROM Employee AS E
JOIN Sales AS S ON E.dept = S.dept
JOIN Types AS T ON S.item = T.item
WHERE T.color = 'Black';
```


Assigment 3.pdf
Results 1

name
Anderson
Lewis
Nelson

3. For each department that has a larger average salary than that of "Stationery" department, find its average salary.

```sql
-- 1.3.3
SELECT dept, AVG(salary)
FROM Employee
GROUP BY dept
HAVING AVG(salary) > (
        SELECT AVG(salary)
        FROM Employee
        WHERE dept = 'Stationery'
    );
```


Assigment 3.pdf
Results 1

dept	AVG(salary)
Cosmetics	13000

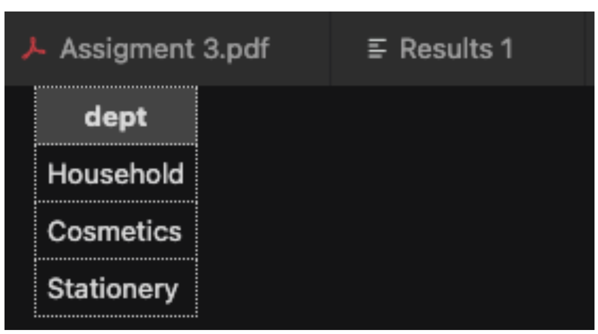
4. Find the number of the departments that have a smaller average salary than that of "Stationery" department.

```sql
-- 1.3.4
SELECT COUNT(DISTINCT dept)
FROM Employee
WHERE dept IN (SELECT dept
                FROM Employee
                GROUP BY dept
                HAVING AVG(salary) < (
                        SELECT AVG(salary)
                        FROM Employee
                        WHERE dept = 'Stationery'
                    ));
```

COUNT(DISTINCT dept)
2

5. Which department pays every of its employees at least 7000?

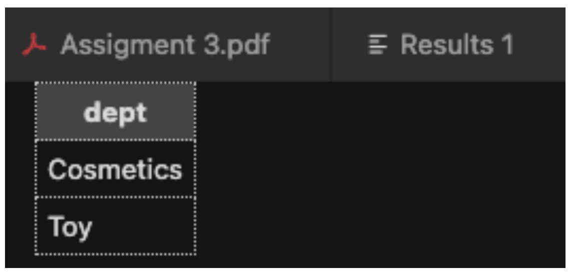
```sql
-- 1.3.5
SELECT dept
FROM Employee
GROUP BY dept
HAVING MIN(salary) >= 7000;
```



dept
Household
Cosmetics
Stationery

6. Which departments sell all items sold by “Cosmetics” department

```sql
-- 1.3.6
SELECT dept
FROM Sales
WHERE item IN (
    SELECT item
    FROM Sales
    WHERE dept = 'Cosmetics'
)
GROUP BY dept
HAVING COUNT(DISTINCT item) = (
    SELECT COUNT(DISTINCT item)
    FROM Sales
    WHERE dept = 'Cosmetics'
);
```



Part2

Question 1

S	A	C	T	D
David	7924	CMPT412	Yasutaka Furukawa	CS
David	7924	CMPT354	Ke Wang	CS
Tom	8850	CMPT354	Ke Wang	CS

Data redundancy:

- **Data is redundancy i.e. `tuple(David,7924)` and `tuple(CMPT354, Ke Wang, CS)`.**
- **“Ke Wang” and “CS” can be figured out by using FD’s `C->T` and `C->D`.**
- **“7924” can be figured out by using FD’s `S->A`**

Update anomaly: if `David` is transfered to `8850`, will we remember ti change each of his tuples

Deletion anomaly: if noone take `CMPT354`, we will loss track of the fact that `tuple(CMPT354, Ke Wang, CS)`

Insertion anomaly: Impossible to insert a student without knowing his/her course schedule.

Question 2

2.1 Find all keys of R with respect to F.

`key = {S,C}` **It is the only key.**

2.2 Test if R in BCNF with respect to F, why?

NO

`C`, `S`, `T` **aren't super key so they violate BCNF.**

2.3 Produce a BCNF decomposition through a series of binary decomposition. For each binary decomposition, tell the FD used for the decomposition and show the FDs holding on the decomposed tables.

- **Pick BCNF violation:** `S->A`
  - **FD used for the decomposition:** `S->A` **:**
    - **R1 =** S^+ **=** `{S, A}` **holds** `S->A`
    - **R2 =** $(R–S^+) \cup S$ **=** `{S,C,T,D}` **holds** `C->T`, `T->D`
- **BCNF violation R2:** `C->T`
  - **FD used for the decomposition:** `C->T`, `T->D`
    - **R3 =** C^+ **=** `{C, T, D}` **holds** `C->T`, `T->D`
    - **R4 =** $(R_2–C^+) \cup C$ **=** `{S, C}`
- **BCNF violation R3:** `T->D`
  - **FD used for the decomposition:** `T->D`, `C->T`
    - **R5 =** T^+ **=** `{T, D}` **holds** `T->D`
    - **R6 =** $(R_3–T^+) \cup T$ **=** `{C, T}` **holds** `C->T`
- **Final result:**
  - **There are 4 relationships (**$R_1, R_2.....$ **is renamed):**
    - **R1 =** `{T, D}` **holds** `T->D`**, T is key**
    - **R2 =** `{S, A}` **holds** `S->A`**, S is key**
    - **R3 =** `{C, T}` **holds** `C->T`**, C is key**
    - **R4 =** `{S, C}`**, {S, C} is key**

2.4 Explain why the decomposed tables produced in 3 is a better representation than the original single table R.

There isn't data redundancy, update anomaly. Insertion anomaly and deletion anomaly.

2.5 Is the final decomposition in 3 dependency-preserving, why

R1 = `{T, D}` holds F1 = `T->D`, T is key

R2 = `{S, A}` holds F2 `S->A`, S is key

R3 = `{C, T}` holds F3 = `C->T`, C is key

R4 = `{S, C}`, {S, C} is key

$F1 \cup F2 \cup F3 = F$. **In other words F1, F2, F3 implies F. So it is dependency-preserving**

2.6 Is the original schema R in 3NF with respect to F, why

No

F = `T->D`, `S->A`, `C->T`.

Key = `{S,C}`

`D`, `A`, `T` is not prime (each of them isn't a member of any key).

`S`, `T`, `C` is not superkey.

2.7 If the answer to 6 is no, produce a 3NF decomposition that is lossless and dependency-preserving.

F = `T->D`, `S->A`, `C->T`.

(3 steps: Split right sides into single attributes, Remove redundant attributes from left sides of FDs, Remove redundant FDs)

minial cover after 3 steps is still `T->D`, `S->A`, `C->T`.

R1 = `{T, D}` holds `T->D`

R2 = `{S, A}` holds `S->A`

R3 = `{C, T}` holds `C->T`

no key is contained in any Ri:

R4 = `{S, C}`

Final result 3NF Decomposition `{TD, SA, CT, SC}`

2.8 Is the decomposition produced in 7 in BCNF?

Yes

$$T^+ \rightarrow R1$$

T is superkey of R1 = `{T, D}` holds `T->D`, D isn't contained in T, it is in BCNF

$$S^+ \rightarrow R2$$

S is superkey of R2 = `{S, A}` holds `S->A`, A isn't contained in S, it is in BCNF

$$C^+ \rightarrow R3$$

C is superkey of R3 = `{C, T}` holds `C->T`, T isn't contained in C, it is in BCNF

For R4 there is no FD so it is in BCNF