1)Convert bookstore.xml into json

XML File

```
<bookstore>
  <book>
    <title>Harry Potter</title>
    <author>J.K. Rowling</author>
    <price>29.99</price>
    <available>true</available>
  </book>
  <book>
    <title>The Hobbit</title>
    <author>J.R.R. Tolkien</author>
    <price>19.99</price>
    <available>false</available>
  </book>
</bookstore>
```

JSON File

```
{
  "bookstore": {
    "book": [
      {
        "title": "Harry Potter",
        "author": "J.K. Rowling",
        "price": 29.99,
        "available": true
      },
      {
        "title": "The Hobbit",
        "author": "J.R.R. Tolkien",
```

```
      "price": 19.99,
      "available": false
    }
  ]
 }
}
```

2) Write a query to give inner join,left outer join,right outer join and full outer join

Employee Table

==========

employee_id	first_name	last_name	department_id
101	ramu	reddy	100
201	soma	Smith	200
301	Metu	John	300
401	dasari	bunny	100

Department Table

=========

department_id	department_name
100	sales
200	HR
300	marketing
400	IT

DESCRIPTION:

INNER JOIN:

An INNER JOIN in MySQL combines rows from two or more tables based on a related column between them. It returns only the rows that have matching values in both tables.

SELECTe.employee_id,e.first_name, e.last_name,e.department_id,d.department_name

FROM Employees e

INNER JOIN Department d ON e.department_id=d.department_id;

employee_id	first_name	last_name	department_id	department_name
101	Ramu	Reddy	100	HR
201	Soma	Smitha	200	Sales
301	Mettu	John	300	IT
401	Dasari	Bunny	100	HR

LEFT OUTER JOIN:

Returns all rows from the left table, and the matched rows from the right table. If no match is found, NULL values are returned for columns from the right table.

SELECT e.employee_id, e.first_name, e.last_name, e.department_id, d.department_name

FROM Employee AS e

LEFT JOIN Department AS d ON e.department_id = d.department_id;

employee_id	first_name	last_name	department_id	department_name
101	Ramu	Reddy	100	HR
201	Soma	Smith	200	Sales
301	Metu	John	300	IT
401	dasari	Bunny	100	HR

RIGHT OUTER JOIN:

Returns all rows from the right table, and the matched rows from the left table. If no match is found, NULL values are returned for columns from the left table.

SELECT e.employee_id, e.first_name, e.last_name, e.department_id, d.department_name

FROM Employee AS e

RIGHT JOIN Department AS d ON e.department_id = d.department_id;

employee_id	first_name	last_name	department_id	department_name
101	Ramu	Reddy	100	sales
201	Soma	Smith	200	HR
301	Metty	John	300	marketing

employee_id	first_name	last_name	department_id	department_name
401	dasari	Bunny	100	sales
NULL	NULL	NULL	400	IT

FULL OUTER JOIN:

Returns all rows when there is a match in either left or right table. If there is no match, the result is NULL on the side that does not have a match

SELECT e.employee_id, e.first_name, e.last_name, e.department_id, d.department_name

FROM Employee AS e

LEFT JOIN Department AS d ON e.department_id = d.department_id

UNION

SELECT e.employee_id, e.first_name, e.last_name, e.department_id, d.department_name

FROM Employee AS e

RIGHT JOIN Department AS d ON e.department_id = d.department_id;

employee_id	first_name	last_name	department_id	department_name
101	Ramu	Reddy	100	HR
201	Soma	Smith	200	Sales
301	Mettu	John	300	IT
401	Dasari	Bunny	100	HR
NULL	NULL	NULL	400	Marketing

3) Write a query to find duplicate records

employee_id	first_name	last_name	email
101	ramu	Reddy	Ramu.reddy@example.com
201	Soma	Smith	Soma.smith@example.com
301	Ramu	Reddy	Ramu.reddy@example.com
401	Dasrai	Bunny	Dasari.bunny@example.com

Find Duplicate records

1) Based on firstName

2) based on email

3) Based on firstname and Last Name

4) Based on firstname and Email

1) Based on firstName:

SELECT first_name, COUNT(*) FROM Employee GROUP BY first_name HAVING COUNT(*) > 1;

first_name	COUNT(*)
John	1

2) based on email

SELECT email, COUNT(*) FROM Employee GROUP BY email HAVING COUNT(*) > 1;

email	COUNT(*)
Ramu.reddy@example.com	1

3) Based on firstname and Last Name

SELECT first_name, last_name, COUNT(*) FROM Employee GROUP BY first_name, last_name HAVING COUNT(*) > 1;

first_name	last_name	COUNT(*)
Ramu	Reddy	1

4) Based on firstname and email

SELECT first_name, email, COUNT(*) FROM Employee GROUP BY first_name, email HAVING COUNT(*) > 1;

first_name	email	COUNT(*)
Ramu	Ramu.reddy@example.com	1