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

```
{
 "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.

→Description

Inner Join:

- An inner join returns only the rows that have matching values in both tables based on the specified condition.
- It selects records that have matching values in both tables' columns being joined.
- If there is no match found, the rows from both tables will be excluded from the result set.
- Inner joins are typically the most common type of join used in SQL queries.

Left Outer Join:

- A left outer join returns all the rows from the left table, and the matching rows from the right table. If there is no match found in the right table, NULL values are included in the result set for the columns from the right table.
- It preserves the unmatched rows from the left table.

Right Outer Join:

- A right outer join returns all the rows from the right table, and the matching rows from the left table. If there is no match found in the left table, NULL values are included in the result set for the columns from the left table.

- It preserves the unmatched rows from the right table.

Full Outer Join:

- A full outer join returns all the rows when there is a match in either the left or right table. If there is no match found, NULL values are included in the result set for the columns from the other table.
- It includes all the rows from both tables, regardless of whether there is a match or not.

→Queries

```
mysql> select empid,first_name,last_name,deptname from employee
inner join department on employee.depid=department.depid

;

+-------+------------+-----------+----------+

| empid | first_name | last_name | deptname |

+-------+------------+-----------+----------+

|     1 | john       | doe       | HR       |

|     2 | jane       | smith     | sales    |

|     3 | mike       | johnson   | IT       |

|     4 | emily      | davis     | HR       |

+-------+------------+-----------+----------+

4 rows in set (0.00 sec)


mysql> select empid,first_name,last_name,deptname from employee left
outer join department on employee.depid=department.

depid;

+-------+------------+-----------+----------+

| empid | first_name | last_name | deptname |

+-------+------------+-----------+----------+

|     1 | john       | doe       | HR       |

|     2 | jane       | smith     | sales    |
```

```
|     3 | mike       | johnson   | IT       |
|     4 | emily      | davis     | HR       |
+-------+------------+-----------+----------+
4 rows in set (0.00 sec)

mysql> select empid,first_name,last_name,deptname from employee
right outer join department on employee.depid=department
.depid;
+-------+------------+-----------+-----------+
| empid | first_name | last_name | deptname  |
+-------+------------+-----------+-----------+
|     4 | emily      | davis     | HR        |
|     1 | john       | doe       | HR        |
|     2 | jane       | smith     | sales     |
|     3 | mike       | johnson   | IT        |
|  NULL | NULL       | NULL      | Marketing |
+-------+------------+-----------+-----------+
5 rows in set (0.00 sec)
```

3)Write a query to find duplicate records

```
mysql> select* from duplicates;
+-------+------------+-----------+-------------------------+
| empid | first_name | last_name | email                   |
+-------+------------+-----------+-------------------------+
|     1 | john       | doe       | john.doe@example.com    |
|     2 | jane       | smith     | jane.smith@example.com  |
|     3 | john       | doe       | john.doe@example.com    |
|     4 | emily      | davis     | emily.davis@example.com |
```

```
+-------+-----------+----------+------------------------+
```

Based on firstname.

```
mysql> select first_name,count(*) FROM duplicates GROUP BY
first_name HAVING COUNT(*) > 1;

+------------+----------+

| first_name | count(*) |

+------------+----------+

| john       |        2 |

+------------+----------+

1 row in set (0.00 sec)
```

Based on email.

```
mysql> select email,count(*) FROM duplicates GROUP BY email HAVING
COUNT(*) > 1;

+------------------------+----------+

| email                  | count(*) |

+------------------------+----------+

| jane.smith@example.com |        2 |

+------------------------+----------+

1 row in set (0.00 sec)
```

Based on firstname,lastname.

```
mysql> select first_name,last_name,count(*) FROM duplicates GROUP BY
first_name,last_name HAVING COUNT(*) > 1;

+------------+-----------+----------+

| first_name | last_name | count(*) |

+------------+-----------+----------+

| john       | doe       |        2 |

+------------+-----------+----------+

1 row in set (0.00 sec)
```

Based on firstname,email.

```
mysql> select first_name,email,count() from employes group by
first_name,email having count()>1;
+------------+----------------------+----------+
| first_name | email                | count(*) |
+------------+----------------------+----------+
| Jhon       | jhon.doe@example.com |        2 |
+------------+----------------------+----------+
1 row in set (0.00 sec)
```