Lap1 postrgesql

1- What is NoSQL?

NoSQL (Not Only SQL) is a category of database management systems that do not use the traditional relational database model. Instead of tables with rows and columns, NoSQL databases use a variety of data models, including document, key-value, wide-column, and graph formats. NoSQL databases are designed to handle large volumes of unstructured or semi-structured data and are optimized for horizontal scaling and high performance.

Key Characteristics of NoSQL:

- **Scalability:** Designed to scale out by distributing data across multiple servers.
- **Flexibility:** Can handle various types of data models (document, key-value, wide-column, graph).
- **Performance:** Optimized for fast read and write operations.
- **Schema-less:** Do not require a fixed schema, allowing for more flexibility in data storage and retrieval.

Examples of NoSQL Databases:

- **Document Store:** MongoDB, CouchDB
- **Key-Value Store:** Redis, DynamoDB
- **Wide-Column Store:** Cassandra, HBase
- **Graph Database:** Neo4j, OrientDB

2- Types of DBMS (Database Management Systems)

1. **Hierarchical DBMS:**
    - **Structure:** Uses a tree-like structure where each record has a single parent and possibly many children.
    - **Example:** IBM Information Management System (IMS).
    - **Use Case:** Early banking systems and telecommunications.
2. **Network DBMS:**
    - **Structure:** Similar to hierarchical DBMS but allows many-to-many relationships in a graph-like structure.
    - **Example:** Integrated Data Store (IDS), IDMS (Integrated Database Management System).
    - **Use Case:** Complex applications like digital asset management and telecommunication networks.
3. **Relational DBMS (RDBMS):**
    - **Structure:** Uses tables (relations) to store data. Each table consists of rows and columns.
    - **Example:** MySQL, PostgreSQL, Oracle, SQL Server.

- **Use Case:** Traditional business applications, including ERP, CRM, and accounting systems.

4. **Object-Oriented DBMS (OODBMS):**
    - **Structure:** Stores data in the form of objects, similar to object-oriented programming.
    - **Example:** ObjectDB, db4o.
    - **Use Case:** Applications requiring complex data representations, such as CAD/CAM, and multimedia applications.
5. **Document-Oriented DBMS (NoSQL):**
    - **Structure:** Stores data in documents, often using formats like JSON or BSON.
    - **Example:** MongoDB, CouchDB.
    - **Use Case:** Content management systems, web applications, and real-time analytics.
6. **Key-Value DBMS (NoSQL):**
    - **Structure:** Stores data as key-value pairs.
    - **Example:** Redis, DynamoDB.
    - **Use Case:** Caching, session management, and real-time data processing.
7. **Wide-Column Store DBMS (NoSQL):**
    - **Structure:** Uses tables, rows, and dynamic columns, designed for large-scale data storage.
    - **Example:** Apache Cassandra, HBase.
    - **Use Case:** Distributed data storage, large-scale data analytics, and time-series data.
8. **Graph DBMS (NoSQL):**
    - **Structure:** Uses graph structures with nodes, edges, and properties to represent and store data.
    - **Example:** Neo4j, OrientDB.
    - **Use Case:** Social networks, recommendation engines, and network topology.

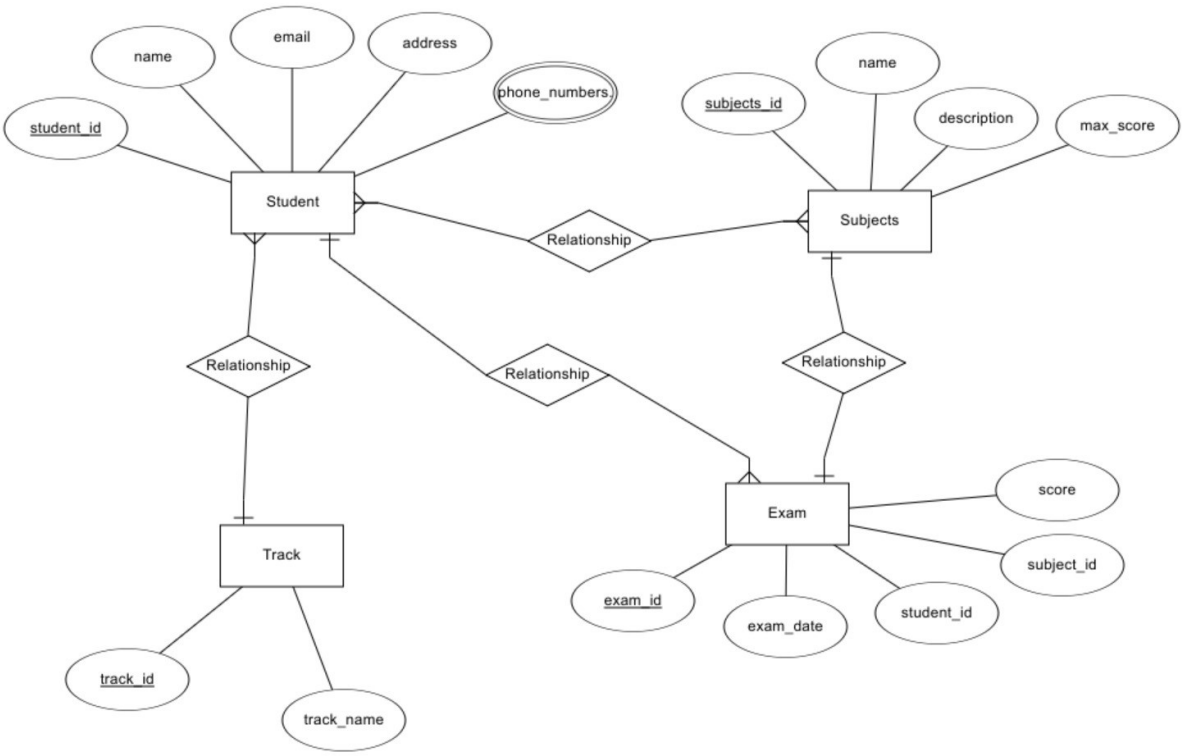
```
postgres=# \c lab1
You are now connected to database "lab1" as user "postgres".
lab1=# \dt
           List of relations
 Schema |     Name     | Type  |  Owner
--------+--------------+-------+----------
 public | courses      | table | postgres
 public | exams        | table | postgres
 public | phonenumbers | table | postgres
 public | students     | table | postgres
 public | studenttrack | table | postgres
 public | tracks       | table | postgres
(6 rows)

lab1=#
```

```
lab1=# SELECT * FROM Tracks;
 track_id | track_name
----------+-------------
        1 | Telecom
        2 | OpenSource
        3 | Java
        4 | Game
        5 | DataScience
(5 rows)

lab1=#
```

```
lab1=# SELECT * FROM Students;
 student_id |  name  |      email       | address
------------+--------+------------------+----------
          1 | helana | helana@gmail.com | 123  St.
          2 | safa   | safa@gmail.com   | 456  St.
          3 | jeje   | jeje@gmail.com   | 789 st.
          4 | David  | david@gmail.com  | 101 St.
          5 | lolo   | lolo@gmail.com   | 202 St.
(5 rows)

lab1=#
```

```
lab1=# SELECT * FROM PhoneNumbers;
 phone_id | student_id |  phone_number
----------+------------+----------------
        1 |          1 | 010123-456-7890
        2 |          1 | 015123-456-7891
        3 |          2 | 011234-567-8901
        4 |          3 | 012345-678-9012
        5 |          4 | 012456-789-0123
        6 |          5 | 010567-890-1234
(6 rows)

lab1=#
```

```
lab1=# SELECT * FROM Exams;
 exam_id | student_id | course_id | exam_date  | score
---------+------------+-----------+------------+-------
       1 |          1 |         1 | 2024-07-01 |    85
       2 |          2 |         2 | 2024-07-02 |    90
       3 |          3 |         3 | 2024-07-03 |    78
       4 |          4 |         4 | 2024-07-04 |    88
       5 |          5 |         5 | 2024-07-05 |    92
       6 |          1 |         5 | 2024-07-10 |    80
       7 |          2 |         1 | 2024-07-11 |    85
       8 |          3 |         4 | 2024-07-12 |    70
       9 |          4 |         3 | 2024-07-13 |    95
      10 |          5 |         2 | 2024-07-14 |    75
(10 rows)

lab1=#
```


name
email
address
phone_numbers
student_id
Student
Relationship
Subjects
subjects_id
name
description
max_score
Relationship
Relationship
Relationship
Track
track_id
track_name
Exam
score
subject_id
exam_id
exam_date
student_id

```
lab1=# SELECT * FROM StudentTrack;
 student_id | track_id
------------+----------
          1 |        1
          2 |        2
          3 |        3
          4 |        4
          5 |        5
(5 rows)

lab1=#
```