# 1B INSY6112 2025

ST10288567

Darsh Somayi

## Question 1

## Database for a Large Social Media Platform

## Database: NoSQL Database

### What is a NoSQL Database?

A NoSQL (Not Only SQL) database is a non-relational database designed to oversee large volumes of unstructured, semi-structured, and structured data (Chodorow, 2013).

Unlike traditional relational databases, NoSQL databases have no schema, allowing for flexible data models that can grow horizontally across dispersed systems. They are designed for peak performance, real-time processing, and large data applications (Chodorow, 2013).

Why Choose NoSQL for a Social Media Platform?

1. Scalability for High Volume of Data:

Every day, social media networks generate large volumes of data, necessitating the use of a horizontally scalable system (Hewitt, 2016).

NoSQL databases, such as MongoDB or Cassandra, are designed for distributed architectures, allowing for seamless scaling to accommodate increasing data loads without sacrificing speed (Hewitt, 2016).

2. Flexibility of Complex and Evolving Data Structures

Social media data encompasses a wide range of content kinds (text, photos, videos, and live streams) as well as interactions. NoSQL databases have dynamic schemas, which allow the platform to store a variety of data formats without requiring strict table designs, making it easier to implement new features (Chodorow, 2013).

3. Real-time data processing provides instant updates.

NoSQL databases thrive in low latency read/write operations, which are necessary for real-time analytics (trending posts, engagement metrics) and user experience. Systems such as Apache Cassandra and Redis are designed for rapid data ingestion and retrieval (Carlson, 2013).

Types of NoSQL Databases Used in Social Media Platforms:

1. Document databases:

   Document databases store data in JSON-like formats (such as BSON in MongoDB), which allows for layered structures. They are best suited for content management, user profiles, and flexible schemas. Queries are optimised for quick retrieval, making them ideal for social media feeds (Chodorow, 2013).

2.Key-Value Stores (e.g., Redis, DynamoDB)

These databases store data in key-value pairs, allowing for extremely quick read/write operations. They are used for caching, session management, and real-time analytics (e.g., calculating likes per minute). Redis, an in-memory key-value store, is commonly used for fast data access (Carlson, 2013).

3.Wide-Column Stores (e.g., Apache Cassandra, HBase)

These databases arrange data in columns rather than rows, allowing for large-scale distributed storage. Cassandra excels at handling time-series data (for example, post engagement metrics) and maintains high availability across numerous servers (Hewitt, 2016).

4.Graph Databases (e.g., Neo4j, Amazon Neptune)

Graph databases, which store data in the form of nodes and links, are suitable for social networks. They efficiently navigate complex interactions, such as locating mutual friends or recommending content (Robinson et al., 2015).

Applying the Three Vs of Big Data to a Social Media Platform:

1.Volume

The platform deals with massive data growth from millions of users generating posts, interactions, and media. NoSQL databases efficiently manage this volume through distributed storage and horizontal scaling (Hewitt, 2016).

2.Variety

Data comes in multiple formats—structured (user profiles), semi-structured (JSON posts), and unstructured (images, videos). NoSQL databases support this diversity without requiring fixed schemas (Chodorow, 2013).

3.Velocity

Real-time interactions (likes, comments, live streams) require rapid data processing. NoSQL databases enable high-speed writes and low latency reads, ensuring instant updates and analytics (Carlson, 2013).

IN Conclusion:

A NoSQL database is the greatest option for this social media network because of its scalability, flexibility, and real-time capabilities. It efficiently processes high-volume, diversified, and fast-moving data while providing a consistent user experience.

**Question 2**

**Entity Relationship Diagram (ERD) for Online Bookstore**

**Below is the ERD in UML notation based on the provided requirements:**

**Entities and Attributes for diagram:**

Customer

Customer (PK)

Full Name

Email

AddressLine1

AddressLine2

Suburb

City

Book

BookID (PK)

Title

Author

Genre

Price

QuantityInStock

Order

OrderID (PK)

OrderDate

Status (Pending/Shipped/Delivered)

TotalPrice

CustomerID (FK)

Payment

PaymentID (PK)

PaymentDate

PaymentAmount

PaymentMethod

OrderID (FK)

Review

ReviewID (PK)

Rating (1-5)

ReviewText

ReviewDate

CustomerID (FK)

BookID (FK)

OrderBook (Junction table to resolve many-to-many between Order and Book)
OrderID (PK, FK)

BookID (PK, FK)

Quantity

Relationships
Customer to Order (1-to-many):

A Customer can place many Orders.

Exactly one Customer places each Order.

Order to Payment (1-to-1):

An Order has exactly one Payment

A Payment is associated with exactly one Order.

Customer to Review (1-to-many):

A Customer can write many Reviews.

Exactly one Customer writes each Review.

Book to Review (1-to-many):

A Book can have many Reviews.

Each Review is for exactly one Book.

Order to Book (many-to-many, resolved via OrderBook junction table):

An Order can contain many Books (with quantities)

A Book can appear in many Orders.

Multiplicities

Customer (1) → (0..*) Order

Order (1) → (1) Payment

Customer (1) → (0..*) Review

Book (1) → (0..*) Review

Order (1) → (1..*) OrderBook → (1) Book

Book (1) → (0..*) OrderBook → (1) Order

Primary Keys

All entities have single-attribute primary keys except OrderBook which has a composite primary key (OrderID + BookID)

Foreign Keys

Order contains CustomerID (FK to Customer)

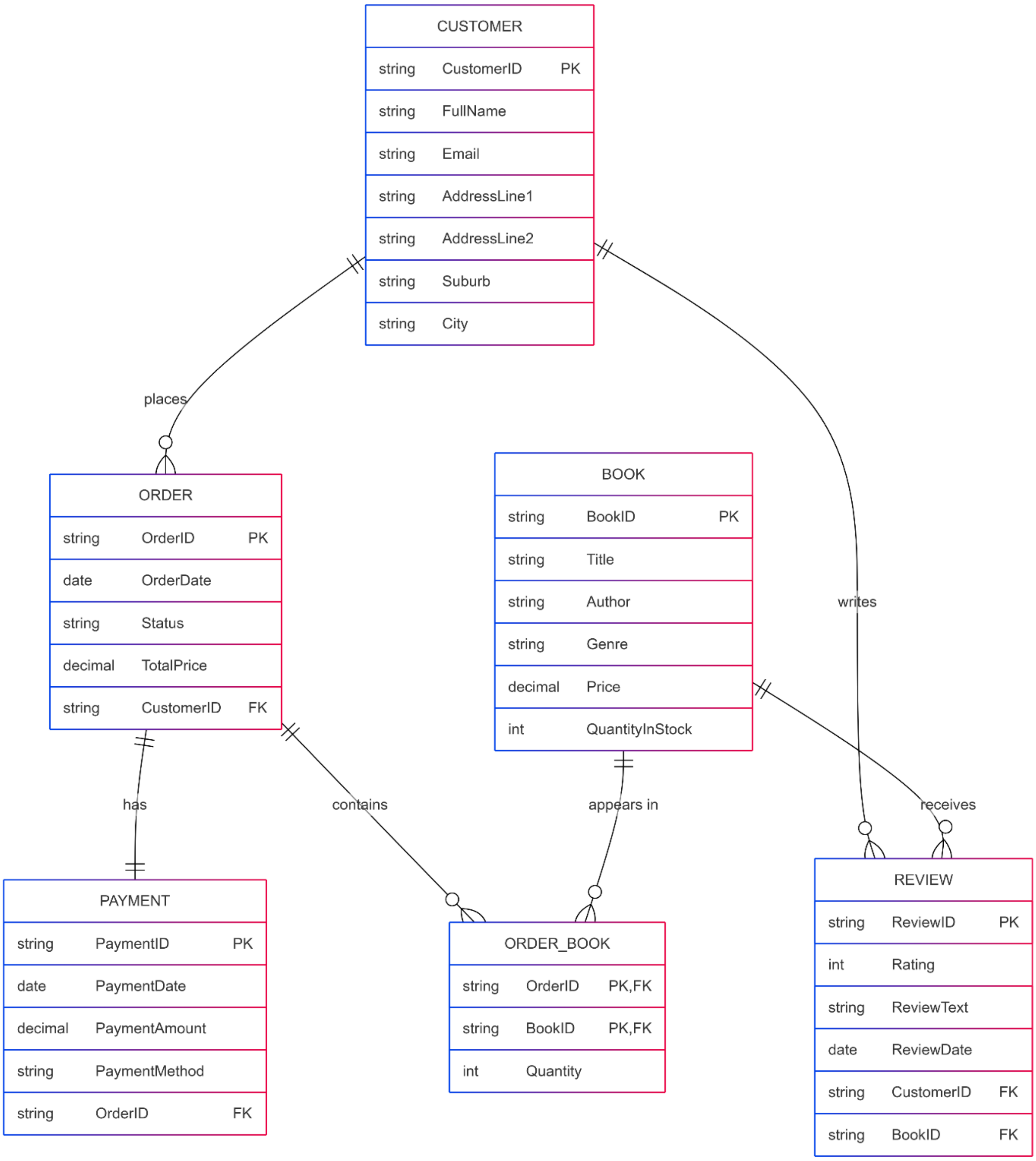
Payment contains OrderID (FK to Order)

Review contains CustomerID (FK to Customer) and BookID (FK to Book)

OrderBook contains OrderID (FK to Order) and BookID (FK to Book)

(Chodorow, 2013).

DIAGRAM:




*Draw.io/ (Accessed: 24 March 2025).*

This diagram meets all the business rules and standards mentioned, including:

Unique identifiers for each entity(Chodorow, 2013).

Effective resolution of many-to-many relationships

Suitable attribute types and limitations. Maintain all specified business rules about linkages and cardinalities (Chodorow, 2013).

## REFERENCE LIST

**References and documentation for the four NoSQL databases mentioned in the report:**

*Chodorow, K. (2013) MongoDB: The definitive guide. 2nd edn. Sebastopol, CA: O'Reilly Media. Available at: https://www.oreilly.com/library/view/mongodb-the-definitive/9781449344689/ (Accessed: 28 March 2025).*

*Carlson, L. (2013) Redis in Action. Shelter Island, NY: Manning Publications. Available at: https://www.manning.com/books/redis-in-action (Accessed: 28 March 2025).*

*Hewitt, E. (2016) Cassandra: The definitive guide. 2nd edn. Sebastopol, CA: O'Reilly Media. Available at: https://www.oreilly.com/library/view/cassandra-the-definitive/9781491933664/ (Accessed: 28 March 2025).*

*Robinson, I., Webber, J. and Eifrem, E. (2015) Graph databases. 2nd edn. Sebastopol, CA: O'Reilly Media. Available at: https://www.oreilly.com/library/view/graph-databases-2nd/9781491930885/ (Accessed: 28 March 2025).*

*Diagram:*

*Free Flowchart Maker and diagrams online (2025) Flowchart Maker & Online Diagram Software. Available at: http://Draw.io/ (Accessed: 24 March 2025).*