Databases

An Introduction

8 February 2024

克明峻德，格物致知

For 10-20 years, spreadsheets is a back-bone IT system for the business.

In fact, they are still being used for internal housekeeping and external data exchange.

Q: It's working, but why databases? Where is the bottleneck?

A: Scalibilities. Business with high volume won't be possible without a database.

Think (also): 'Last-Mile Deliveries'.

Excel is Microsoft's best consumer product.

Satya Nadella, Microsoft CEO

Course Overview

Focus Areas:

- In-Depth SQL Queries
- Principles of Database Design
- ETL Techniques

Tools:

- MySQL, Google Colaboratory, DBDiagram, ERDPlus

Learning Outcomes:

- Mastery in SQL Query Processing and Effective Data Management
- Skills in Scalable Database Design and Optimisation

What is Database?

- **Definition:** A Database is a structured collection of data, stored electronically for efficient access and management.
- **Role:** Essential in a wide range of applications, from web services to complex data analytics.
- **Data Types:** Diverse, including text, numbers, multimedia.

Database Management Systems (DBMS)

Key Functions and Role:

- **Definition:** DBMS is software that facilitates efficient data management in databases.
- **Primary Functions:**
  - Data Organization and Security: Organizing data for efficient access and safeguarding against unauthorized access.
  - Maintenance and Accessibility: Ensuring data integrity through backup and recovery, and supporting multi-user access.
  - Query Processing: Executing data retrieval and manipulation.

DBMS Examples:

- **Commercial:** Oracle, IBM DB2, Microsoft SQL Server.
- **Open-Source:** MySQL, PostgreSQL.

Importance in Data Management:

- DBMSs are essential for secure, efficient, and reliable data handling in diverse applications.

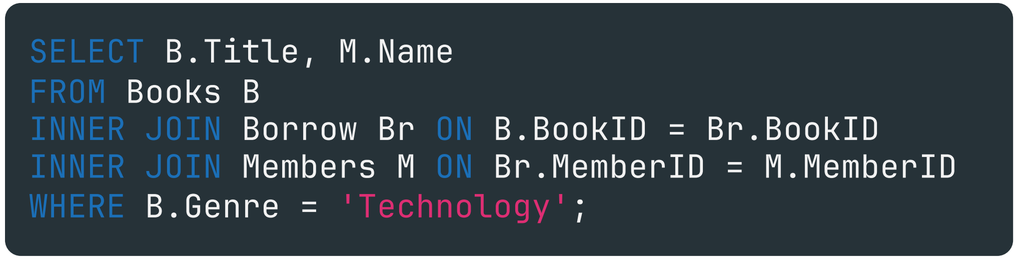
The Data Model Concept

Overview:

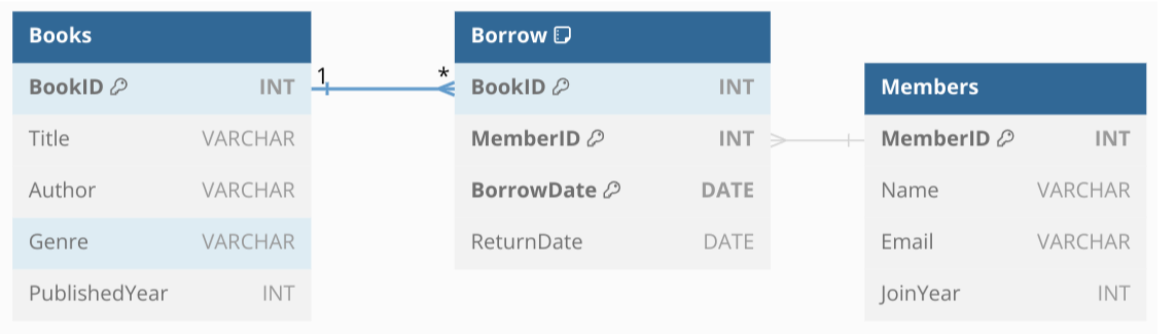
- Data models are blueprints for structuring and managing data in a database.

Key Elements:

- **Schema:** Outlines the structure of the entire database, including table designs and relationships.
  - *Example:* Schema for a library database with tables for 'Books', 'Members', and 'Borrow'.
  - *Linking Tables:* 'Borrow' connects 'Books' and 'Members', mapping book borrowings to members via BookID and MemberID.
- **Instance:** Actual data stored in the database at any given time.
- **Query Language (SQL):** Used for data retrieval and manipulation.
  - *SQL Example:*



```sql
SELECT B.Title, M.Name
FROM Books B
INNER JOIN Borrow Br ON B.BookID = Br.BookID
INNER JOIN Members M ON Br.MemberID = M.MemberID
WHERE B.Genre = 'Technology';
```



Books	
BookID	INT
Title	VARCHAR
Author	VARCHAR
Genre	VARCHAR
PublishedYear	INT

Borrow	
BookID	INT
MemberID	INT
BorrowDate	DATE
ReturnDate	DATE

Members	
MemberID	INT
Name	VARCHAR
Email	VARCHAR
JoinYear	INT

Relational Data Model

Understanding the Relational Data Model:

- The relational data model is a framework for organizing data in tables (rows and columns).

Key Characteristics:

- **Set Semantics:** Ensures uniqueness and no duplication in rows.
- **Typed and Static Attributes:** Each column has a defined data type for consistency and integrity.
- **Flat Table Structure:** Non-hierarchical, straightforward data organization.

Schema Example with Data:

- Books Table Schema:
  - Columns: BookID (INT), Title (VARCHAR), Author (VARCHAR), Genre (VARCHAR), PublishedYear (INT).
- Example Data:

```
| BookID | Title                      | Author          | Genre       | PublishedYear |
|--------|----------------------------|-----------------|-------------|---------------|
| 101    | Journey Through SQL        | A. Coder        | Technology  | 2015          |
| 102    | The History of Databases   | D. Base         | Education   | 2018          |
| 103    | Adventures in Coding       | P. Programmer   | Fiction     | 2020          |
| ...    | ...                        | ...             | ...         | ...           |
```

Detailed Exploration of Relational Model Structure

Table Structure:

- **Tables (Relations):** Represent collections of related data entries.
- **Columns (Attributes):** Define the data's properties, like name, type, and category.
- **Rows (Tuples):** Each row indicates a single record or data entry.

Example: Books Table:

- **Schema:** BookID (INT), Title (VARCHAR), Author (VARCHAR), Genre (VARCHAR), PublishedYear (INT).
- Data Representation:

```
| BookID | Title                      | Author          | Genre        | PublishedYear |
|--------|----------------------------|-----------------|--------------|---------------|
| 101    | Journey Through SQL        | A. Coder        | Technology   | 2015          |
| 102    | The History of Databases   | D. Base         | Education    | 2018          |
| 103    | Adventures in Coding       | P. Programmer   | Fiction      | 2020          |
| ...    | ...                        | ...             | ...          | ...           |
```

Detailed Exploration of Relational Model Structure

Table Structure:

- **Tables (Relations):** Represent collections of related data entries.
- **Columns (Attributes):** Define the data's properties, like name, type, and category.
- **Rows (Tuples):** Each row indicates a single record or data entry.

Example: Books Table:

- **Schema:** BookID (INT), Title (VARCHAR), Author (VARCHAR), Genre (VARCHAR), PublishedYear (INT).
- Data Representation:

```
| BookID | Title                      | Author         | Genre       | PublishedYear |
|--------|----------------------------|----------------|-------------|---------------|
| 101    | Journey Through SQL        | A. Coder       | Technology  | 2015          |
| 102    | The History of Databases   | D. Base        | Education   | 2018          |
| 103    | Adventures in Coding       | P. Programmer  | Fiction     | 2020          |
| ...    | ...                        | ...            | ...         | ...           |
```

Detailed Exploration of Relational Model Structure

Table Structure:

- **Tables (Relations):** Represent collections of related data entries.
- **Columns (Attributes):** Define the data's properties, like name, type, and category.
- **Rows (Tuples):** Each row indicates a single record or data entry.

Example: Books Table:

- **Schema:** BookID (INT), Title (VARCHAR), Author (VARCHAR), Genre (VARCHAR), PublishedYear (INT).
- Data Representation:

```
| BookID | Title                     | Author         | Genre       | PublishedYear |
|--------|---------------------------|----------------|-------------|---------------|
| 101    | Journey Through SQL       | A. Coder       | Technology  | 2015          |
| 102    | The History of Databases  | D. Base        | Education   | 2018          |
| 103    | Adventures in Coding      | P. Programmer  | Fiction     | 2020          |
| ...    | ...                       | ...            | ...         | ...           |
```

Detailed Exploration of Relational Model Structure

Table Structure:

- **Tables (Relations):** Represent collections of related data entries.
- **Columns (Attributes):** Define the data's properties, like name, type, and category.
- **Rows (Tuples):** Each row indicates a single record or data entry.

Example: Books Table:

- **Schema:** BookID (INT), Title (VARCHAR), Author (VARCHAR), Genre (VARCHAR), PublishedYear (INT).
- Data Representation:

```
| BookID | Title                     | Author          | Genre         | PublishedYear |
|--------|---------------------------|----------------|-------------|---------------|
| 101    | Journey Through SQL       | A. Coder        | Technology    | 2015          |
| 102    | The History of Databases  | D. Base         | Education     | 2018          |
| 103    | Adventures in Coding      | P. Programmer   | Fiction       | 2020          |
| ...    | ...                       | ...             | ...           | ...           |
```

Order Does Not Matter in Sets

Understanding Set Semantics in Relational Databases

- **Key Concept:** In the relational model, the order of rows in a table (set) is not significant. What matters is the data each row contains.

- Original Books Table:

```
| BookID | Title                      | Author          | Genre        | PublishedYear |
|--------|----------------------------|-----------------|--------------|---------------|
| 101    | Journey Through SQL        | A. Coder        | Technology   | 2015          |
| 102    | The History of Databases   | D. Base         | Education    | 2018          |
| 103    | Adventures in Coding       | P. Programmer   | Fiction      | 2020          |
| ...    | ...                        | ...             | ...          | ...           |
```

- Reordered Books Table:

```
| BookID | Title                      | Author          | Genre        | PublishedYear |
|--------|----------------------------|-----------------|--------------|---------------|
| 103    | Adventures in Coding       | P. Programmer   | Fiction      | 2020          |
| 101    | Journey Through SQL        | A. Coder        | Technology   | 2015          |
| 102    | The History of Databases   | D. Base         | Education    | 2018          |
| ...    | ...                        | ...             | ...          | ...           |
```

- Note: Even though the order of rows is different, the dataset remains unchanged in terms of content and meaning.

Set Semantics in the Relational Model

Key Points

- **Set Semantics Principle:** Ensures each row in a relational table is unique, crucial for data integrity.
- **Violation Example:** Demonstrating duplication in the 'Books' table.
  - Books Table Data (with Violation):

```
| BookID | Title                     | Author       | Genre       | PublishedYear |
|--------|---------------------------|--------------|-------------|---------------|
| 101    | Adventures in Coding      | P. Programmer| Fiction     | 2020          |
| 102    | Mysteries of the Universe | S. Hawking   | Science     | 2019          |
| 103    | The History of Databases  | D. Base      | Education   | 2018          |
| 103    | The History of Databases  | D. Base      | Education   | 2018          |
```

Note: Duplicate entry for "The History of Databases" illustrates violation.

Types and Static Attributes in the Relational Model

Understanding Data Types and Constraints:

- Significance: Ensures data integrity and query accuracy in relational databases.
- Issue: Incorrect data types lead to inconsistencies and operational challenges.

Example of Data Type Violation:

- The 'PublishedYear' for "Future of Data" is incorrectly entered as a string, demonstrating the violation of the attribute type constraint.

```
| BookID | Title                     | Author         | Genre        | PublishedYear        |
|--------|---------------------------|----------------|--------------|----------------------|
| 101    | Journey Through SQL       | A. Coder       | Technology   | 2015                 |
| 102    | The History of Databases  | D. Base        | Education    | 2018                 |
| 103    | Adventures in Coding      | P. Programmer  | Fiction      | 2020                 |
| 104    | Future of Data            | I. Analyst     | Non-Fiction  | 'Twenty Twenty-One'  |
```

Flat Table Structure in the Relational Model

Explanation of the Principle:

- In relational databases, each table should represent a single entity, avoiding nested or hierarchical structures.
- Flat tables enhance data management simplicity and query performance.

Violation Example:

- Incorrect 'Books' Table Structure:
  - Demonstrates violation with 'Editions' column containing nested data.

```
| Title                       | PublishedYear | Editions                         |
|---------------------------- |---------------|----------------------------------|
| Adventures in Coding        | 2020          | Edition 1: {2020, ISBN 12345} |
|                             |               | Edition 2: {2021, ISBN 67890} |
| Mysteries of the Universe | 2019          | Edition 1: {2019, ISBN 23456} |
| The History of Databases  | 2018          | Edition 1: {2018, ISBN 34567} |
```

'Editions' column inappropriately combines multiple pieces of data.

Best Practices:

- Employ normalization to maintain data integrity and ease of access.

Structured Query Language (SQL)

SQL Introduction:

- SQL is the primary language for relational database management and manipulation.
- Focuses on specifying 'what' data is needed, rather than 'how' to retrieve it.

IBM SEQUEL ← SQL

(Structure English Query Language)

SQL Queries

SELECT Queries:

- Overview: Fundamental tool for data retrieval in SQL.
- Function: Enables precise extraction of data from database tables.

Example and Explanation

- SQL Query:

```
SELECT *
FROM Books;
```

- **Purpose:** Fetching all data from the 'Books' table to illustrate the simplicity and effectiveness of SQL queries.
- Output Table:

```
| BookID | Title                      | Author          | Genre         | PublishedYear |
|--------|----------------------------|-----------------|---------------|---------------|
| 101    | Journey Through SQL        | A. Coder        | Technology    | 2015          |
| 102    | The History of Databases   | D. Base         | Education     | 2018          |
| 103    | Adventures in Coding       | P. Programmer   | Fiction       | 2020          |
| ...    | ...                        | ...             | ...           | ...           |
```

More on SQL SELECT Queries

Data Projection with SQL's SELECT Clause:

- The SELECT clause in SQL specifies the columns to be retrieved, enabling focused and efficient data extraction from tables.

Source Table: Books:

```
| BookID | Title                       | Author          | Genre         | PublishedYear |
|--------|-----------------------------|-----------------|---------------|---------------|
| 101    | Journey Through SQL         | A. Coder        | Technology    | 2015          |
| 102    | The History of Databases    | D. Base         | Education     | 2018          |
| 103    | Adventures in Coding        | P. Programmer   | Fiction       | 2020          |
| ...    | ...                         | ...             | ...           | ...           |
```

SQL Query:

```sql
SELECT Title, Author
FROM Books;
```

Output Table:

```
| Title                       | Author          |
|-----------------------------|-----------------|
| Journey Through SQL         | A. Coder        |
| The History of Databases    | D. Base         |
| Adventures in Coding        | P. Programmer   |
| ...                         | ...             |
```

Data Representation

Different data models can represent the same data set, each suited specific types of analysis and queries.

Case Study: Public Transportation Network

- Graph Representation:

```
+-------------+     Blue Line     +-----------+     Blue Line     +-------------+
|             | ----------------> |           | ----------------> |             |
|   Main St   |                   | Downtown  |                   | Central     |
|             | <---------------- |           | <---------------- | Park        |
+-------------+     Red Line      +-----------+     Red Line      +-------------+
```

- Relational Model:
  - *Tables:* Stations, Routes and StationRoutes.

Stations

```
| StationID | StationName  |
|-----------|--------------|
| 1         | Main St      |
| 2         | Downtown     |
| 3         | Central Park |
```

Routes

```
| RouteID | RouteName     |
|---------|---------------|
| A       | Blue Line     |
| B       | Red Line      |
```

StationRoutes

```
| StationID | RouteID      |
|-----------|--------------|
| 1         | A            |
| 2         | A            |
| 2         | B            |
| 3         | B            |
```

Choosing the Right Data Model

Key Models:

- **Relational Model:** Ideal for structured data, requiring strong integrity.
- **Document-Oriented (NoSQL):** Flexible schemas, suited for unstructured data.
- **Graph-Based:** Effective for complex relationships and network analysis.
- **Key-Value:** Fast access, useful for caching and real-time operations.

mongoDB MySQL

neo4j redis

Selection Factors:

- Data Type and Structure: Nature and complexity of the data.
- Performance and Scalability: Efficiency in processing and growth handling.
- Application Use Case: Specific requirements based on the industry or task.

DBMS Technology Evolution

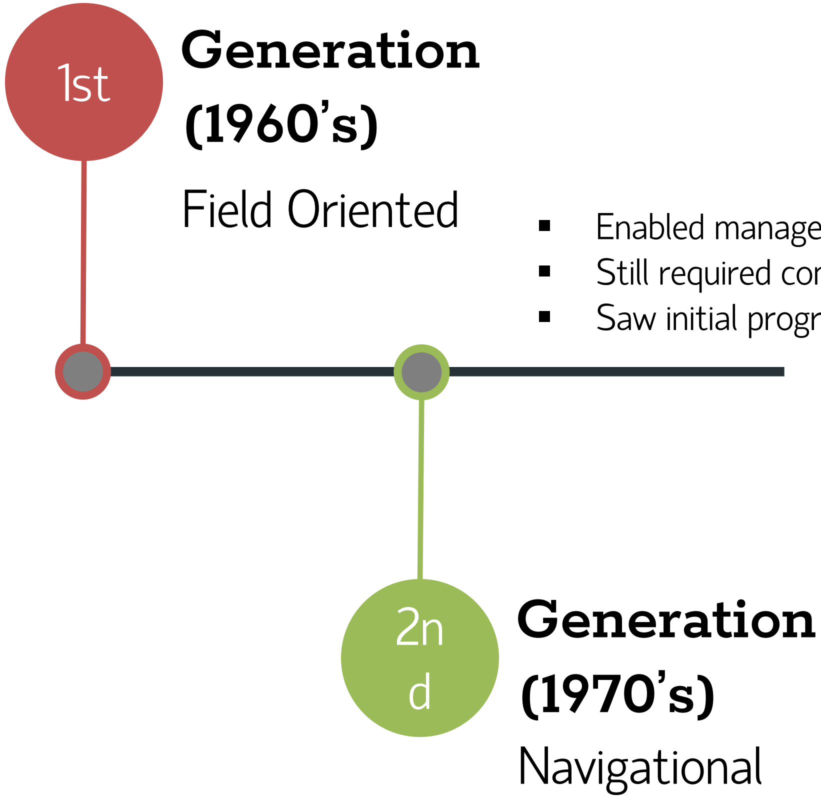
1st

Generation (1960’s)

Field Oriented

- Supported sequential and random searching of files.
- Required users to write detailed computer programs for data access.
- Lacked standardization and interoperability.

DBMS Technology Evolution



1st **Generation (1960’s)**

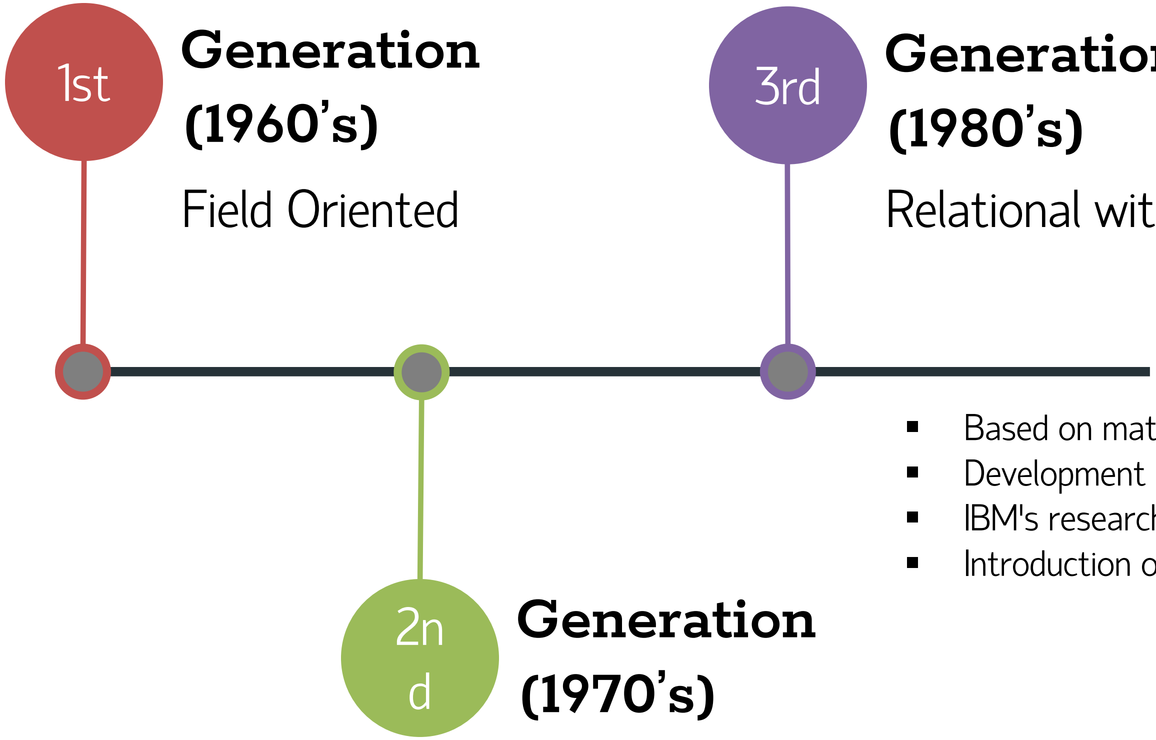
Field Oriented

2nd **Generation (1970’s)**

Navigational

- Enabled management of multiple entity types and complex relationships.
- Still required computer programming for data access.
- Saw initial progress in the development of database standards.

DBMS Technology Evolution



1st

Generation (1960’s)

Field Oriented

2n
d

Generation (1970’s)

Navigational

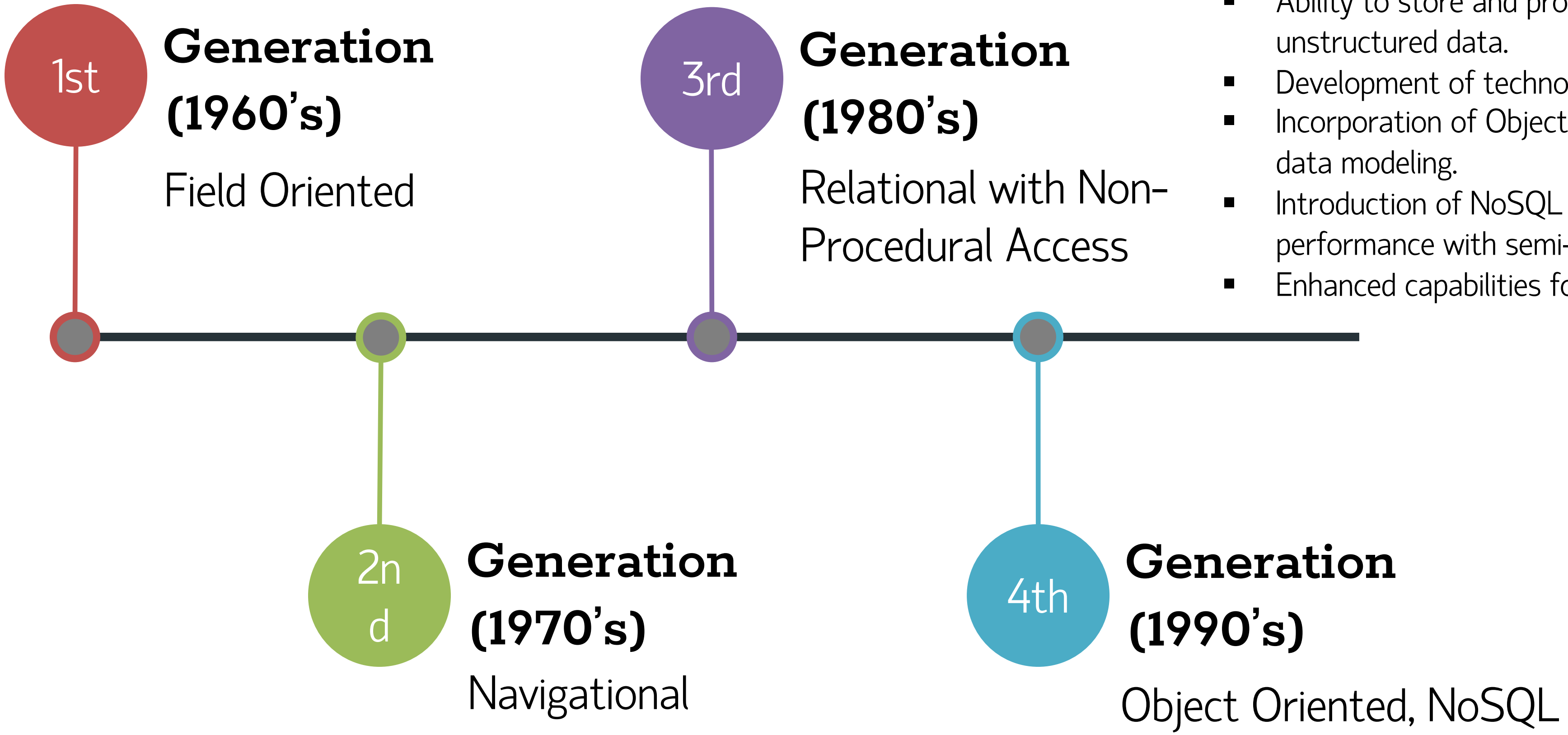
3rd

Generation (1980’s)

Relational with Non-Procedural Access

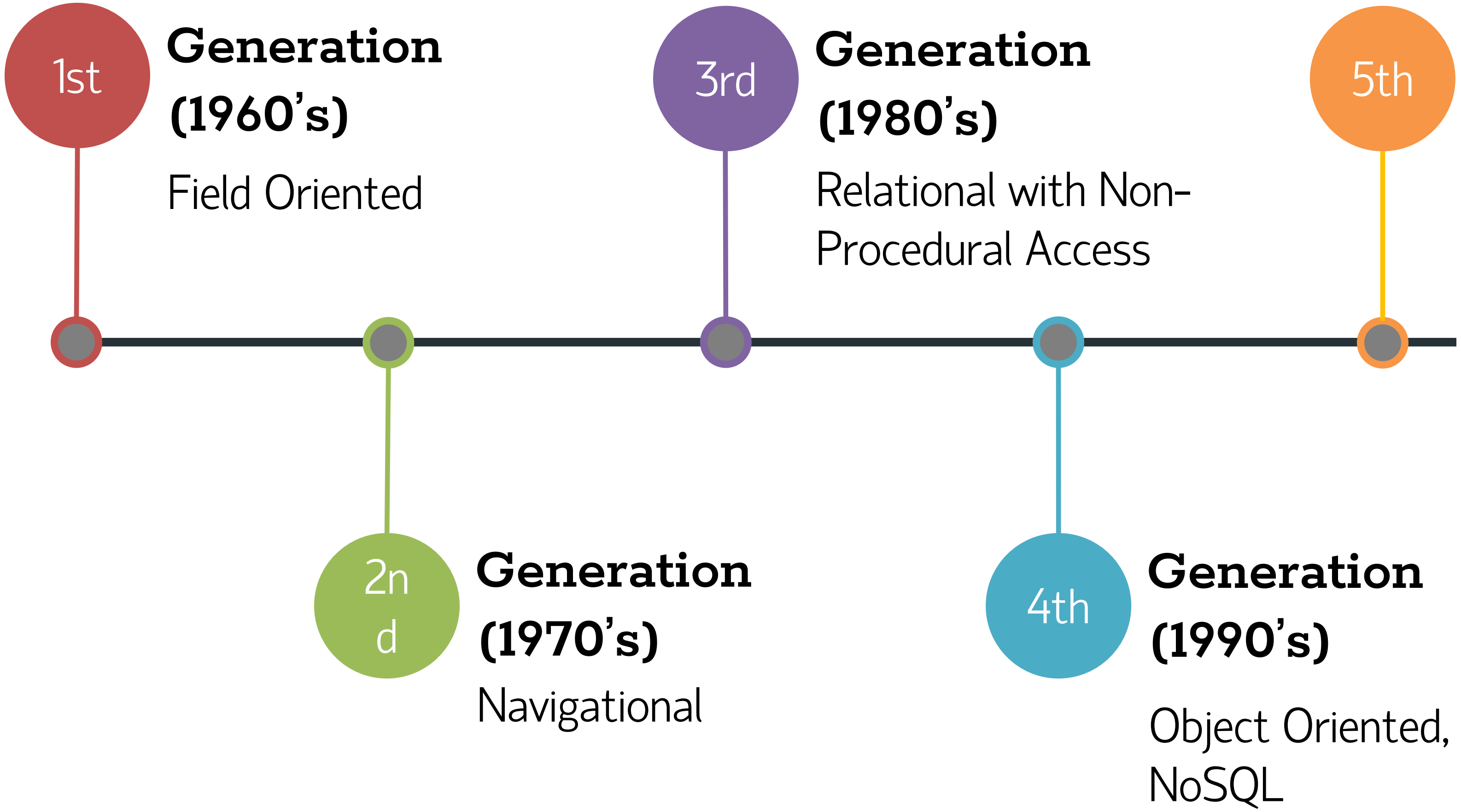
- Based on mathematical set theory and relational algebra.
- Development of query optimization technologies.
- IBM's research played a pivotal role in commercializing relational database technology.
- Introduction of SQL (Structured Query Language) for database interaction.

DBMS Technology Evolution



- Extension of database technology beyond traditional boundaries.
- Emergence of distributed databases and data warehousing.
- Ability to store and process non-traditional data types, including unstructured data.
- Development of technologies for efficient handling of Big Data.
- Incorporation of Object-Oriented features for more complex data modeling.
- Introduction of NoSQL databases for high scalability and performance with semi-structured and unstructured data.
- Enhanced capabilities for web data management and publishing.

DBMS Technology Evolution



1st **Generation (1960's)**

Field Oriented

2n
d **Generation (1970's)**

Navigational

3rd **Generation (1980's)**

Relational with Non-Procedural Access

4th **Generation (1990's)**

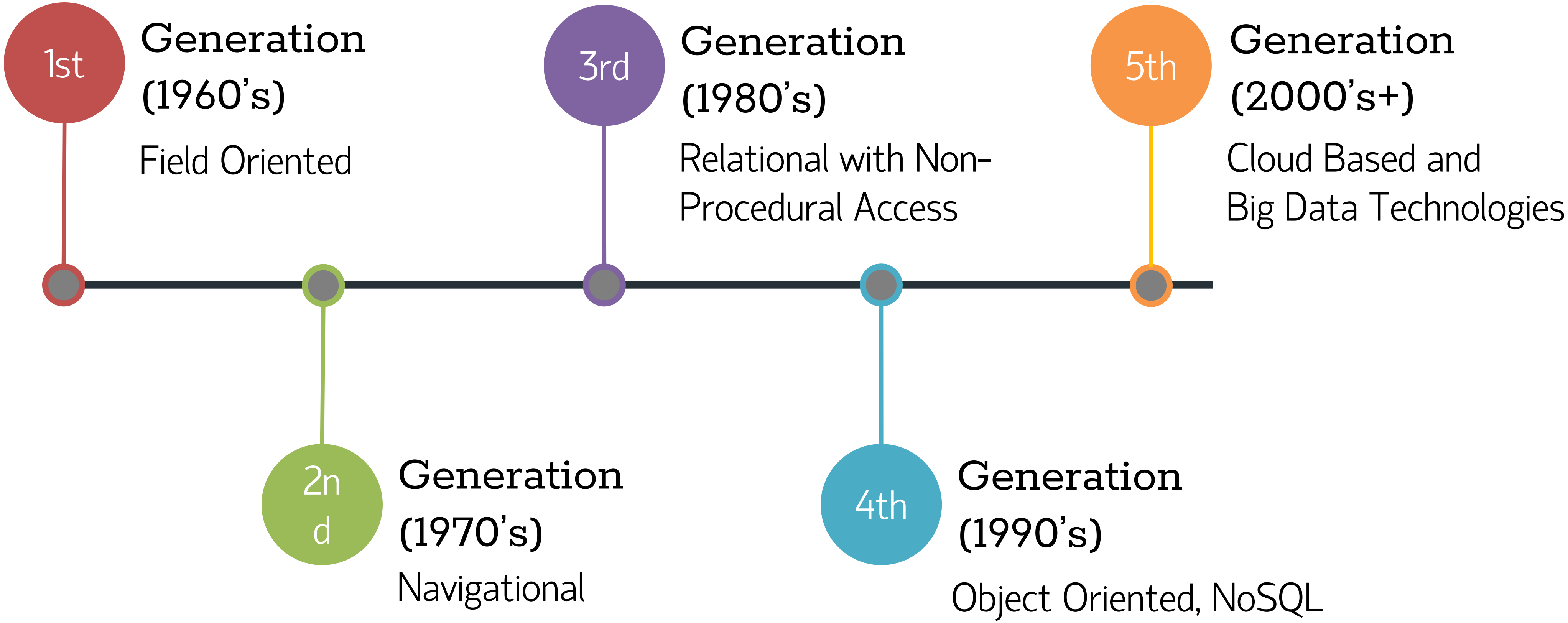
Object Oriented, NoSQL

5th **Generation (2000's+)**

Cloud Based and
Big Data Technologies

- Emergence of cloud-based database services, offering scalability, flexibility, and cost-efficiency.
- Growth in Big Data technologies and analytics.
- Advancements in real-time processing and streaming data.
- Integration of machine learning and AI capabilities for advanced data analytics and prediction.

DBMS Technology Evolution



1st Generation (1960's)
Field Oriented

2n
d Generation (1970's)
Navigational

3rd Generation (1980's)
Relational with Non-Procedural Access

4th Generation (1990's)
Object Oriented, NoSQL

5th Generation (2000's+)
Cloud Based and
Big Data Technologies

Recap and Key Takeaways

- The relational model offers a structured, efficient way to store, retrieve, and manipulate data. Its tabular format, governed by set theory and predicate logic, not only simplifies data handling but also enhances data integrity and consistency.
- DBMS technologies provides a foundation for management of long-term memory of organisations. DBMS enables daily operational tasks and supports short- and long-term decision makings in business.

Example SQL Recap:

```
SELECT Title, Author
FROM Books;
```

- Output emphasizes SQL's role in fetching relevant data.

Preparing for SQL Basics:

- Next Lecture Preview:
  - Dive deeper into **basic SQL queries and functions**.
  - Understand **Primary Key and Foreign Key**.

How might relational databases and SQL evolve with emerging technologies like AI?