# Advanced Relational Model Concepts

## Introduction

In this module, you have learned about advanced relational concepts such as functional dependencies, multi-valued dependencies, and candidate keys.

Review your knowledge:

- Functional dependency (FD): This refers to a relationship between attributes where the value of one attribute uniquely determines the value of another.
- Multi-valued dependency (MVD): This describes a relationship between attributes where one attribute determines a set of possible values for another.
- Candidate key: This denotes a minimal set of attributes that uniquely identifies each row in a relation.

Now, in this reading, let's apply the concepts learned in this module to a real-world example of a database.

## Objectives

After completing this reading, you will be able to evaluate your knowledge of Advanced relational model concepts.

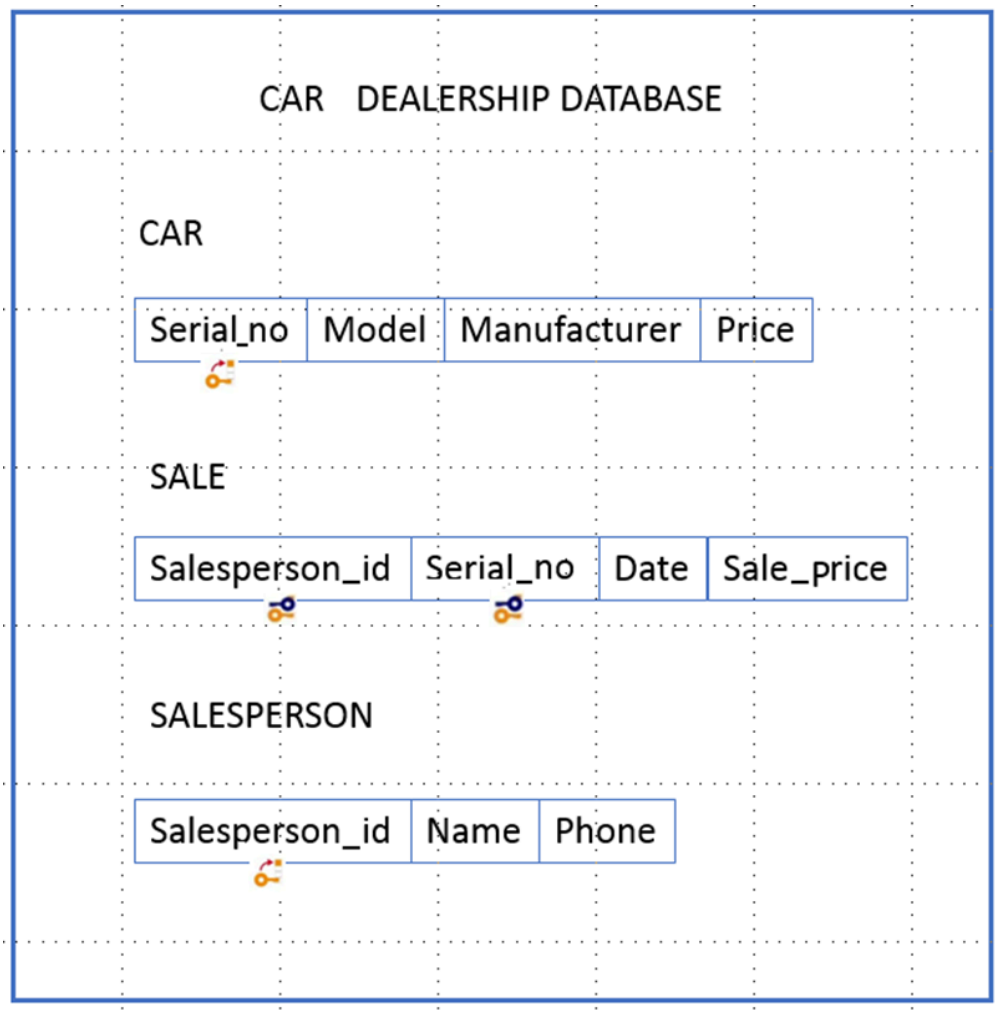
Here you are going to:

- Apply advanced relational concepts like functional dependencies, multi-valued dependencies, and candidate keys to the "Car Dealership" database schema.
- Identify constraints within the schema based on these concepts.
- Understand the impact of these concepts on data integrity and manipulation.

## Exercise

In this exercise, we will work on a relational database schema called Car Dealership, designed to keep track of automobile sales in a car dealership.

**Schema diagram for the Car Dealership relational database:**




**Relational instance of SALE:**

| Salesperson_id | Serial_no | Date | Sale_price |
|---|---|---|---|
| 10001 | 1we4ds87 | 12/03/2020 | $ 10,000.00 |
| 10005 | d63jw3ty | 12/03/2020 | $ 5,000.00 |
| 10009 | sy63bjd1 | 13/03/2020 | $ 25,000.00 |
| 10001 | k2k4edr8 | 13/03/2020 | $ 49,000.00 |
| 10051 | w3r334ac | 13/03/2020 | $ 8,000.00 |

Now, let's go through some questions based on the above database schema of Car Dealership and the relational instance of SALE:

1. Identify FDs in the Car Dealership schema:

   A. Analyze each pair of attributes in each relation (Car, Sale, Salesperson, Customer).

   B. For each pair, consider if the value of one attribute always determines the value of the other.

   C. List all identified FDs for each relation.

   ▼ Answer

   Car:

   - Serial_no → (Model, Manufacturer, Price)
   - Model → Manufacturer

   Sale:

   - Serial_no → Date
   - Serial_no → Sale_Price

   Salesperson:

   - Salesperson_id → Name
   - Salesperson_id → Phone

2. Explore MVDs:

   A. Consider if any attribute in the schema determines a set of possible values for another.

   B. For example, does "Car Model" determine a set of possible values for "Sale Price"?

   C. List any identified MVDs for the schema.

   ▼ Answer

   No MVDs are explicitly identified in the given schema.

3. Determine candidate keys:

A. Analyze each relation and identify any subset of attributes that uniquely identifies each row.

B. Remember, a candidate key must not contain any redundant attributes.

C. List all identified candidate keys for each relation.

▼ Answer

- Car: Serial_no
- Sale: Serial_no
- Salesperson: Salesperson_id

4. Discuss the implications:

   A. How do the identified FDs and MVDs impact data integrity and manipulation in the schema?

   B. Could any data inconsistencies arise due to violating these constraints?

   C. How do candidate keys affect query optimization and data retrieval?

   ▼ Answer

   A. Data Integrity: The identified FDs and candidate keys help ensure data integrity by preventing inconsistencies:

   - Changing Serial_no automatically updates dependent attributes in Car.
   - Serial_no and Date prevents duplicate sales and ensures association with correct car and salesperson.
   - Unique keys (Serial_No, Salesperson_id ) guarantee distinct, identifiable entities.

   Data Manipulation: FDs guide proper data updates. Modifying VIN requires cascading changes to dependent attributes.

   B. Potential Inconsistencies due to Constraint Violations:

   - Incorrect data updates: Forgetting to update dependent attributes when modifying a determining attribute (e.g., changing Serial_no without updating Model) can lead to inconsistencies.

- Duplicate data: if Serial No doesn't determine Price, multiple entries with the same car could have different prices.
- Inaccurate queries: A salesperson selling a non-existent car if Salesperson_id doesn't determine Serial_no.

C. Effects on query optimization and data retrieval:

- Impact of candidate keys: Unique candidate keys (Serial_no for Car, Serial_no for Sale, etc.) significantly improve query performance by creating efficient indexing mechanisms.
- Fast lookups: Queries using candidate keys can quickly locate specific rows without scanning the entire table, resulting in faster data retrieval.
- Reduced redundancy: Candidate keys eliminate redundant identifiers, resulting in smaller data storage requirements and potentially faster table scans for non-indexed searches.