Delta-exam-topics-Data-Modelling

1. Conceptual Data Model

A **conceptual model** is a high-level design that shows how different entities relate to each other. It does **not** include technical details like data types or keys.

Example:
Imagine you're designing a university database. A conceptual model would include:

- **Entities**: Student, Course, Professor
- **Relationships**: A student **enrolls in** a course, a professor **teaches** a course

🔹 Think of it as a rough sketch or blueprint of the system before adding technical details.

2. Physical Data Model

A **physical model** is the **detailed** implementation of the database, including table structures, columns, and data types.

Example:
For a **Student** entity, the physical model may look like this in SQL:

```
CREATE TABLE Student (
    StudentID INT PRIMARY KEY,
    Name VARCHAR(100),
    Age INT,
    Email VARCHAR(255) UNIQUE
);
```

🔹 The physical model ensures the database is optimized for **storage and performance**.

3. Surrogate Key

A **surrogate key** is a unique, system-generated identifier (usually a number). It is **not** derived from real-world data.

Example:

- Instead of using **email** as the primary key, we use a **StudentID** (1, 2, 3…).
- If a student changes their email, it won't affect the database structure.

🔹 It's useful for large databases where natural keys (like email or phone numbers) can change.

4. ERWIN (Data Modeling Tool)

ERWIN is a tool for designing databases visually using **Entity-Relationship Diagrams (ERD)**.

5. Softwares Used for Data Modeling

There are several tools used for **designing databases visually**:

✅ **ERWIN** → Most popular tool for ER diagrams
✅ **Microsoft Visio** → Used for conceptual designs
✅ **Lucidchart** → Web-based modeling tool
✅ **MySQL Workbench** → Used for designing and managing MySQL databases

6. Data modelling types

Data modeling is classified into three levels:

1. **Conceptual Data Model** – Focuses on *what* the system contains.
2. **Logical Data Model** – Defines *how* data should be structured.
3. **Physical Data Model** – Describes *how* data is stored in a specific database.---

1️⃣ Conceptual Data Model

- High-level representation of business entities, attributes, and relationships.
- Created by business analysts and data architects.
- Independent of hardware, software, or database technologies.
- Helps stakeholders understand data at a business level.

Example:

- Entities: **Customer** and **Product**
- Attributes:
  - Customer: `Customer_ID`, `Name`, `Email`
  - Product: `Product_ID`, `Name`, `Price`
- Relationship: **A customer purchases a product.**

Characteristics:
✅ Business-oriented, not technical
✅ No details about tables, columns, or keys
✅ Provides a common vocabulary for stakeholders

2️⃣ Logical Data Model

- Defines the structure of data elements and relationships in detail.
- Still independent of the database management system (DBMS).
- Adds data attributes, data types, and normalization rules.

Example:

- Tables: `Customer`, `Product`, `Sales`
- Data Types:

- `Customer_ID` → Integer (Primary Key)
- `Name` → Varchar(50)
- `Price` → Decimal(10,2)

- Relationship:
  - `Sales` table connects `Customer` and `Product`.

Characteristics:

✅ More detailed than the conceptual model
✅ Includes data types and normalization rules
✅ Helps in refining business rules before database implementation

3️⃣ Physical Data Model

- Database-specific representation of the logical data model.
- Defines tables, columns, constraints, indexes, triggers, etc.
- Developed for a specific DBMS (MySQL, PostgreSQL, SQL Server, etc.).

Example:

- **Table: `Customer`**

```
CREATE TABLE Customer (
    Customer_ID INT PRIMARY KEY,
    Name VARCHAR(50) NOT NULL,
    Email VARCHAR(100) UNIQUE
);
```

- **Table: `Sales`**

```
CREATE TABLE Sales (
    Sale_ID INT PRIMARY KEY,
    Customer_ID INT,
    Product_ID INT,
    Sale_Date DATE,
    FOREIGN KEY (Customer_ID) REFERENCES Customer(Customer_ID),
    FOREIGN KEY (Product_ID) REFERENCES Product(Product_ID)
);
```

Characteristics:

✅ Specific to a DBMS (includes SQL scripts)

✅ Defines indexes, constraints, and storage details

✅ Helps developers implement the actual database

Comparison Table

Feature	Conceptual Model	Logical Model	Physical Model
Purpose	High-level business view	Detailed structure and relationships	Implementation in a specific DBMS
Who Creates It?	Business Analysts, Data Architects	Data Architects, Analysts	DBAs, Developers
Focus	Entities, Attributes, Relationships	Data types, Normalization, Keys	Tables, Indexes, Constraints
Independence	Independent of DBMS	Independent of DBMS	Specific to DBMS