**1. User-Defined Data Types (UDDT)**

**What is a User-Defined Data Type?**

* A **User-Defined Data Type** (UDDT) is a custom data type created by the user based on existing system data types.
* It allows you to **standardize data formats and rules** across tables and databases.
* Helps improve **consistency, readability, and maintainability** of database schemas.
* You can define **constraints** and **rules** on the UDDT to enforce data integrity.

**Why do we use User-Defined Data Types?**

* To **enforce consistent data format** across multiple tables (e.g., phone numbers, zip codes).
* To **improve code clarity** by giving meaningful names to complex or commonly used types.
* To **reduce errors** caused by inconsistent datatype definitions.

**When do we use User-Defined Data Types?**

* When you have **repeated columns** with the same data type and constraints across tables.
* When you want to **apply custom rules or defaults** on data types.
* When working on large projects where **standardization of data types** is important.

**When NOT to use User-Defined Data Types?**

* When data types are used in only one place.
* When you need **complex validation logic** better handled by constraints or triggers.
* When you want to avoid dependency issues (changes to UDDTs affect all tables using them).
* For ad-hoc or one-off columns.

**Syntax:**

CREATE TYPE type\_name

FROM base\_system\_data\_type;

**2. User-Defined Tables (Table Types)**

**What is a User-Defined Table Type?**

* A **User-Defined Table Type** is a custom table structure defined by the user.
* It can be used as a **parameter to stored procedures.**
* Enables passing **multiple rows of data** into stored procedures.
* Useful for **table-valued parameters (TVPs)** to improve performance and code clarity.

**Why do we use User-Defined Table Types?**

* To pass **sets of rows** (tables) as parameters to stored procedures.
* To avoid multiple round trips between application and database for bulk data.
* To simplify **batch processing** of data in SQL Server.
* To create **structured and reusable table formats** for input parameters.

**When do we use User-Defined Table Types?**

* When you want to send **multiple rows of data** to a stored procedure.
* When implementing **bulk inserts or updates** from applications.
* When you want to **encapsulate table structure** for reuse across multiple procedures.
* When you need to maintain **strong typing for table-valued parameters**.

**When NOT to use User-Defined Table Types?**

* For **single row parameters** (use scalar types instead).
* When performance requirements are minimal (simple queries).
* When your database or application does not support table-valued parameters (SQL Server 2008 or later required).

**Syntax:**

CREATE TYPE type\_name AS TABLE

(

column\_name1 data\_type [ NULL | NOT NULL ],

column\_name2 data\_type [ NULL | NOT NULL ],

);