# Valid Data Domain and Invalid Data Domain

## 1-Valid Data Domain

**# Source**: **Chatgpt (GPT-5-Standard) at [8/15/2025]**

The **valid data domain** is simply the **range of input values** that the system is **supposed to accept and process correctly** according to its requirements.

### **In other words**

**# Source**: **Chatgpt (GPT-5-Standard) at [8/15/2025]**

* It’s the set of **all possible valid inputs** that produce **expected, correct behavior**.
* Anything **outside** that range is **invalid data** (and should be rejected or trigger an error).

### **Example 1 – Age Input Field**

**# Source**: **Chatgpt (GPT-5-Standard) at [8/15/2025]**

**Rule:** Age must be between 18 and 60 (inclusive).

* **Valid data domain:** 18 → 60
* **Invalid data domain:** Below 18 or above 60

### **Example 2 – Password Length Rule**

**# Source**: **Chatgpt (GPT-5-Standard) at [8/15/2025]**

**Rule:** Password must be between 8 and 16 characters.

* **Valid data domain:** Length = 8, 9, …, 16
* **Invalid data domain:** Length < 8 or > 16

### ***✅ Where It Is Used***

**# Source**: **Chatgpt (GPT-5-Standard) at [8/23/2025]**

1. ***Input Validation***
   * *Checking user inputs against allowed ranges or formats.*
   * *Example: Age field must be between 0–120.*
2. ***Test Design Techniques***
   * ***Equivalence Partitioning (EP)****: Creating test cases for valid partitions (within the valid domain).*
   * ***Boundary Value Analysis (BVA)****: Testing at the edges of the valid domain.*
3. ***Form and UI Validation***
   * *Preventing incorrect data entry on websites or applications.*
4. ***Database Constraints***
   * *Ensuring inserted data falls within allowed domain (e.g., numeric limits, date formats).*
5. ***API Testing***
   * *Sending valid domain values to verify correct responses.*
6. ***Security Testing***
   * *To prevent attacks like SQL Injection by validating inputs against allowed domains.*

## 2-Invalid Data Domain

**# Source**: **Chatgpt (GPT-5-Standard) at [8/15/2025]**

The **invalid data domain** is the **range of input values** that the system **should NOT accept** because they are **outside the allowed or expected limits** defined in the requirements.

### **In other words**

**# Source**: **Chatgpt (GPT-5-Standard) at [8/15/2025]**

* It’s the set of **all possible invalid inputs** that should trigger **error messages, rejections, or default handling**.
* Testing this domain helps ensure the system can handle incorrect inputs gracefully and securely.

### **Example 1 – Age Input Field**

**# Source**: **Chatgpt (GPT-5-Standard) at [8/15/2025]**

**Rule:** Age must be between 18 and 60 (inclusive).

* **Valid data domain:** 18 → 60
* **Invalid data domain:**
  + **Below** 18 (e.g., 0, 15, 17)
  + **Above** 60 (e.g., 61, 75, 100)

### **Example 2 – Postal Rates**

**# Source**: **Chatgpt (GPT-5-Standard) at [8/15/2025]**

**Rule:** Letter weight must be **up to 100g**.

* **Valid data domain:** 1g → 100g
* **Invalid data domain:**
  + **Zero or negative** weights (e.g., -5g, 0g)
  + **Overweight** letters (>100g, e.g., 101g, 250g)

### **✅ Where It’s Used**

**# Source**: **Chatgpt (GPT-5-Standard) at [8/23/2025]**

1. **Negative Testing**
   * To verify the system **rejects invalid inputs** and displays proper error messages.
   * Example: Entering -10 for age to confirm the system prevents saving.
2. **Form Validation**
   * To ensure fields accept only valid values and give feedback for invalid data.
   * Example: Entering an invalid email (userexample.com) should trigger a validation error.
3. **API Input Validation**
   * Testing API endpoints by sending **out-of-range or malformed data** to confirm proper error codes (e.g., 400 Bad Request).
   * Example: Sending a string instead of a numeric value for an ID field.
4. **Boundary Value Analysis (BVA)**
   * Testing inputs **just outside the valid domain boundaries** to check if the system enforces limits.
   * Example: If valid age is 0–120, test -1 and 121.
5. **Security Testing**
   * Using invalid inputs to simulate **attacks** (e.g., SQL injection, script injection).
   * Example: Inputting '; DROP TABLE users; -- to test SQL injection protection.