# 🧪 Types of Performance Testing

Performance testing is not a one-size-fits-all activity. It encompasses **various specialized types**, each answering specific questions about system performance under different conditions. The choice of test depends on:

* 🎯 Testing goals
* 📊 Anticipated user patterns
* ⚠️ Risk profile of the application

**1️⃣ Load Testing**

**🔍 Definition:**  
Simulates expected concurrent user load or transaction volume under **normal operating conditions**.

**🎯 Key Objectives:**

* Measure response times under average and peak load.
* Verify system stability and resource utilization at normal capacity.
* Identify potential bottlenecks with typical user concurrency.

**💡 Example:**  
Simulating 5,000 users browsing and purchasing on an e-commerce website during a regular sale day to verify the site remains stable, responsive, and accurate.

**2️⃣ Stress Testing**

**🔍 Definition:**  
Pushes the system beyond its designed capacity to evaluate **robustness, stability, and error handling** under extreme conditions.

**🎯 Key Objectives:**

* Discover the system’s **breaking point**.
* Observe system **recovery behavior** after failure.
* Identify data corruption risks or error handling gaps under high stress.
* Evaluate system behavior under **resource exhaustion**.

**💡 Example:**  
Simulating 10,000–50,000+ users trying to book tickets for a popular concert to observe where and how the system fails, and whether it recovers gracefully.

**3️⃣ Endurance Testing (Soak Testing)**

**🔍 Definition:**  
Applies a **continuous load over a prolonged period** (e.g., 24–48 hours) to uncover long-term issues like memory leaks or resource leaks.

**🎯 Key Objectives:**

* Detect memory leaks and improper resource management.
* Check for gradual performance degradation over time.
* Validate system stability during **long-running operations**.

**💡 Example:**  
Running 3,000 concurrent users on a banking app continuously for 48 hours to ensure consistent memory usage, uninterrupted database connectivity, and stability.

**4️⃣ Spike Testing**

**🔍 Definition:**  
Evaluates system behavior when exposed to **sudden, sharp increases and decreases in load** over short time frames.

**🎯 Key Objectives:**

* Assess handling of **abrupt user surges**.
* Observe system **response and recovery**.
* Identify failures during **rapid load shifts**.

**💡 Example:**  
A news site receiving a spike from 1,000 to 15,000 users after breaking news. A spike test simulates such jumps repeatedly to check for failure points and recovery time.

**5️⃣ Volume Testing**

**🔍 Definition:**  
Assesses performance when the system handles **large volumes of data**, particularly focusing on database efficiency and file processing.

**🎯 Key Objectives:**

* Evaluate database **query response times** on large datasets.
* Identify performance issues related to **data volume** (e.g., indexing).
* Test efficiency of backend data processing logic.

**💡 Example:**  
Populating a reporting tool’s database with billions of records to test whether historical sales reports still run within acceptable timeframes.

**6️⃣ Scalability Testing**

**🔍 Definition:**  
Tests the system’s ability to **scale up (more power)** or **scale out (more servers)** to handle increasing loads efficiently.

**🎯 Key Objectives:**

* Measure system performance as resources are scaled.
* Determine if performance improves linearly or degrades with scale.
* Provide data for **capacity planning** and infrastructure investment.

**💡 Example:**  
A SaaS product growing from 10,000 to 100,000 users — scalability testing helps determine how many new servers are needed to maintain consistent performance.

**7️⃣ Benchmark Testing**

**🔍 Definition:**  
Compares the performance of a system against **industry standards** or **internal baselines**.

**🎯 Key Objectives:**

* Establish **reference metrics** (baseline performance).
* Evaluate performance of **new builds or releases** against previous versions.
* Validate system compliance with **SLA/contractual obligations**.
* Support **vendor or third-party evaluations**.

**💡 Example:**  
A financial platform benchmarks its core transaction engine to ensure response time stays under 2 seconds for 95% of transactions, as per SLA commitments.