# **Day 33 - 13 September 2025**

**Document Name:**Day 33 - hmuvvala@ - Hari Gopal Muvvala

### **1. Partition Key + Sort Key**

* A DynamoDB table can use a **composite primary key**:
  + **Partition Key (HASH)** → decides the physical partition where the item is stored.
  + **Sort Key (RANGE)** → allows multiple items with the same partition key, ordered by the sort key.
* Example: ID as partition key, Dept as sort key.
* Enables queries like: *fetch all employees with ID = 1001, sorted by department*.

### **2. Query Operation**

* Used to retrieve items based on primary key values.
* Requires at least the **partition key**.
* Can use conditions on the **sort key** (e.g., begins\_with, between).
* More efficient than Scan because it directly targets partitions.

### **3. Scan Operation with Filters**

* Reads every item in the table.
* Can apply **FilterExpression** to only return items that match conditions.
* Example: return only items where Dept = IT.
* Less efficient for large tables compared to Query.

### **4. Update Expressions**

* UpdateExpression allows modifying specific attributes in an item.
* Example:
  + SET Address = :newAddress
  + REMOVE attribute
  + ADD valueToNumericAttr :increment
* Prevents overwriting the entire item.

### **5. Delete Operations**

* **DeleteItem** → remove a specific record based on key.
* **DeleteTable** → remove the entire table structure and data.
* Both are irreversible actions.

### **6. Use Cases of Partition + Sort Key**

* **Order system** → Partition Key = CustomerID, Sort Key = OrderDate.
* **Employee system** → Partition Key = EmployeeID, Sort Key = Department.
* **IoT data** → Partition Key = DeviceID, Sort Key = Timestamp.

### **7. Best Practices**

* Choose partition keys that distribute traffic evenly to avoid hot keys.
* Use sort keys for time-based or category-based queries.
* Prefer **Query** over **Scan** when possible to reduce cost and improve performance.