## Question 1: What is Redux, and why is it used in React applications? Explain the core concepts of actions, reducers, and the store.

**Ans :** is a state management library used in React (and other JavaScript apps) to manage the application's data in a centralized way.

In React, data is usually passed from parent to child components using props.
However, when your app grows large, managing and sharing state between many components becomes difficult — that's where Redux helps.

It provides a single source of truth (a central store) for the entire application's state, making data flow predictable and easier to debug.

## Why Redux is Used in React

- To manage global state (data shared across multiple components)
- To make state predictable and easier to debug
- To maintain consistency of data throughout the app
- To help with time-travel debugging (see previous states)
- To make testing and scaling easier

Core Concepts of Redux

1. Actions

- Actions describe what you want to do — they are plain JavaScript objects.
- Every action has a type (a string describing the action) and sometimes data (called *payload*).

## • Question 2: How does Recoil simplify state management in React compared to Redux?

**Ans :** What is Recoil?

Recoil is a state management library for React created by Facebook. It helps manage both local and global state easily — and feels more natural to use inside React components than Redux.

### How Recoil Simplifies State Management Compared to Redux

| Feature / Concept | Redux | Recoil |
|---|---|---|
| **Setup Complexity** | Needs store, reducers, actions, dispatch | Very simple — no reducers or actions |
| **Boilerplate Code** | A lot (actions, reducers, constants) | Minimal — only atoms and selectors |
| **Integration with React** | Uses external store connected by <Provider> | Feels like part of React, uses hooks |
| **State Sharing** | Global state only | Works for both local & global state |
| **Learning Curve** | Steeper (requires understanding of reducers, middleware) | Easier (uses familiar React patterns) |
| **Async Data (API calls)** | Uses middleware like thunk or saga | Built-in support with async selectors |

Core Concepts of Recoil

1. Atoms

- Smallest units of state — like React's useState, but shareable across components.
- If multiple components use the same atom, all stay in sync automatically.