WEB APP ARCHITECTURES: MULTI-TIER (2-TIER, 3-TIER) & MVC

Slides created by Manos Papagelis

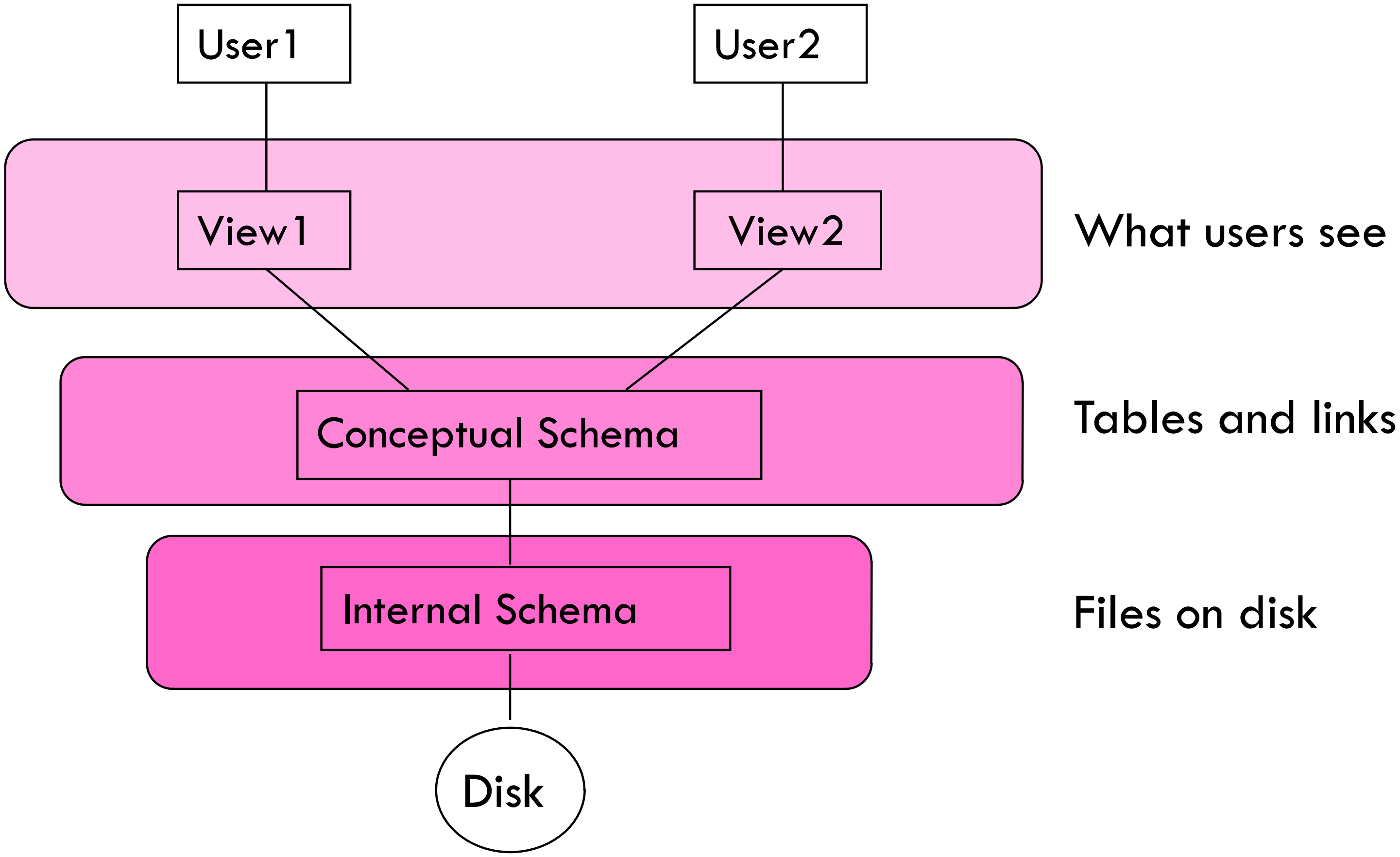
Based on materials by Marty Stepp, M. Ernst, S. Reges, D. Notkin, R. Mercer, R. Boswell,Wikipedia

Overview

- Data Independence in Relational Databases
- N-tier Architectures
- Design Patterns
- The MVC Design Pattern

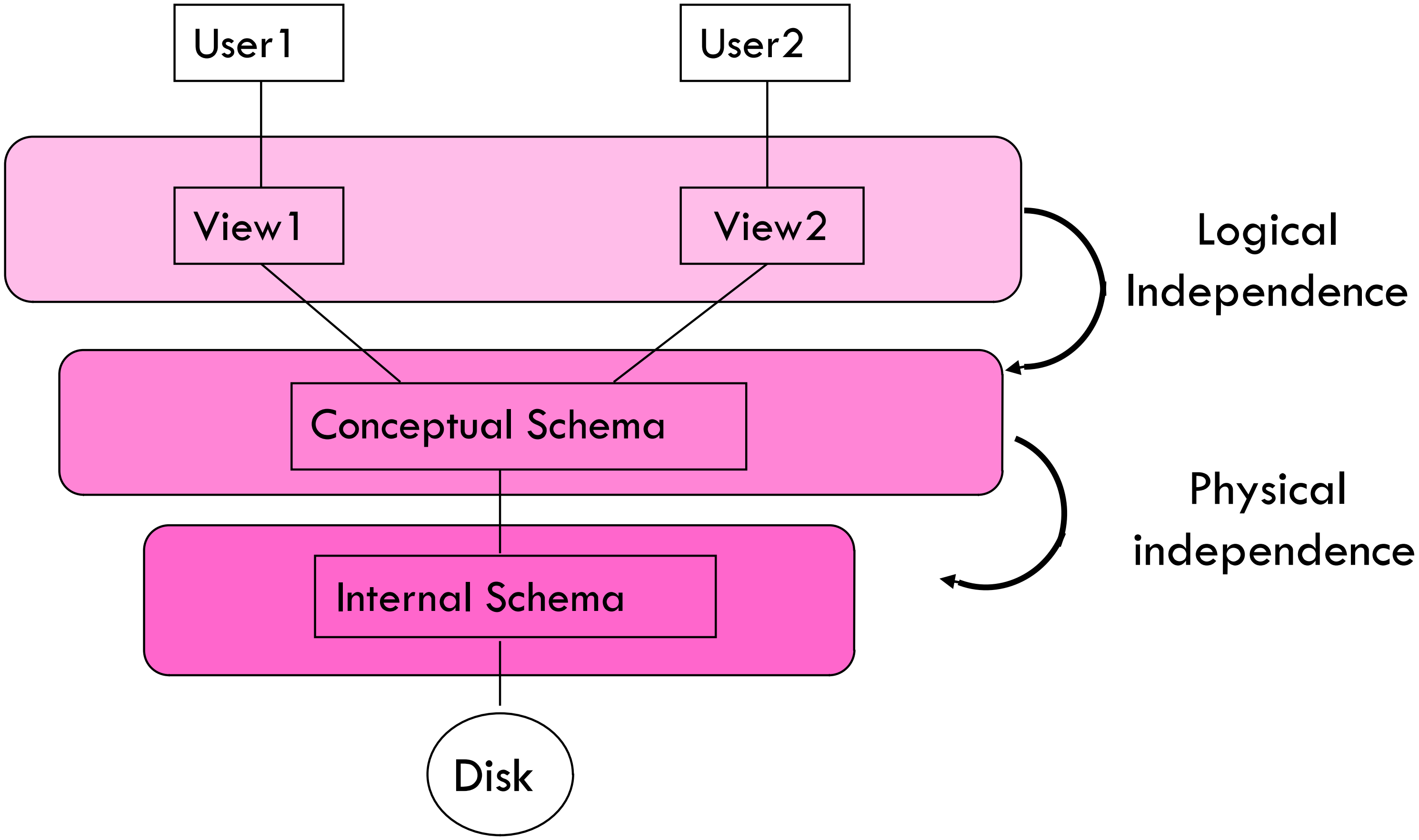
Data Independence in Rel. DBMS

Database Architecture With Views



Each level is independent of the levels below

Logical and Physical Independence



Each level is independent of the levels below

Data Independence

- **Logical Independence:** The ability to change the logical schema without changing the external schema or application programs
  - Can add new fields, new tables without changing views
  - Can change structure of tables without changing view

- **Physical Independence:** The ability to change the physical schema without changing the logical schema
  - Storage space can change
  - Type of some data can change for reasons of optimization

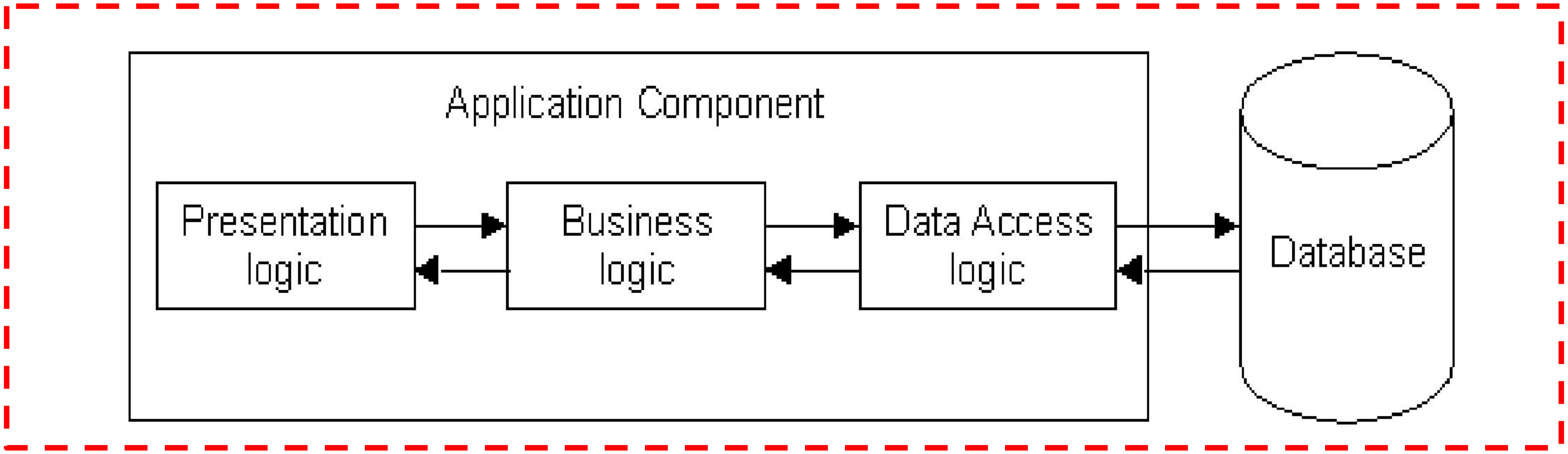
LESSON: Keep the VIEW (what the user sees) independent of the MODEL (domain knowledge)

N-tier architectures

Significance of “Tiers”

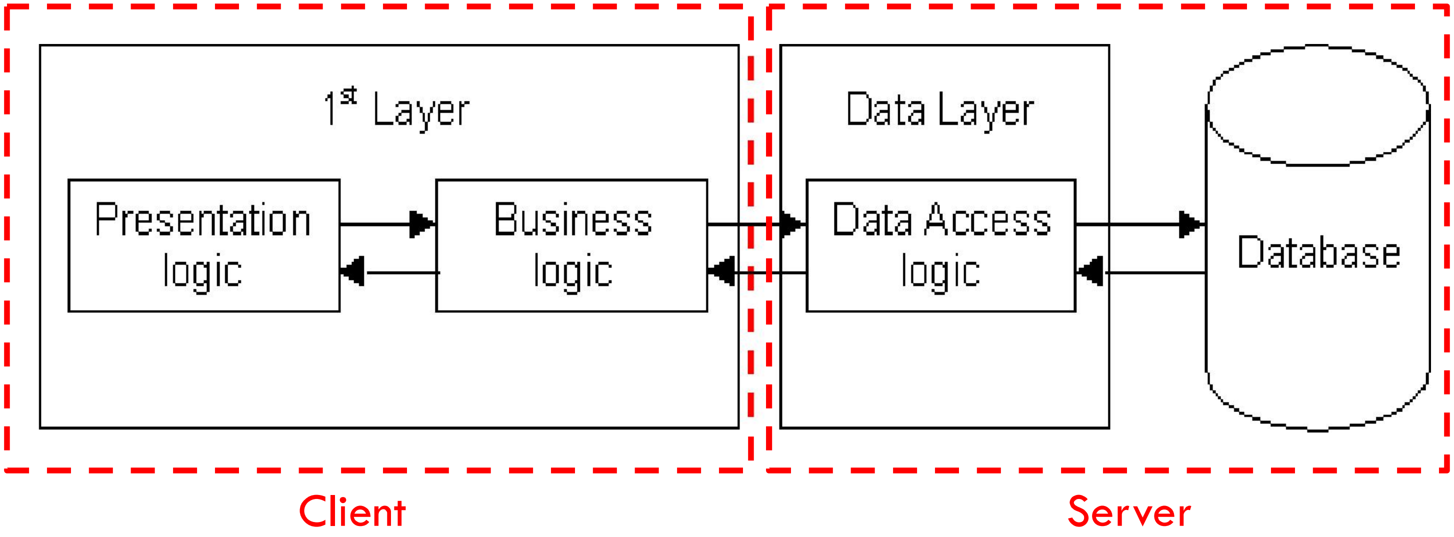
- N-tier architectures have the same components
  - Presentation
  - Business/Logic
  - Data
- N-tier architectures try to separate the components into different tiers/layers
  - Tier: physical separation
  - Layer: logical separation

1-Tier Architecture



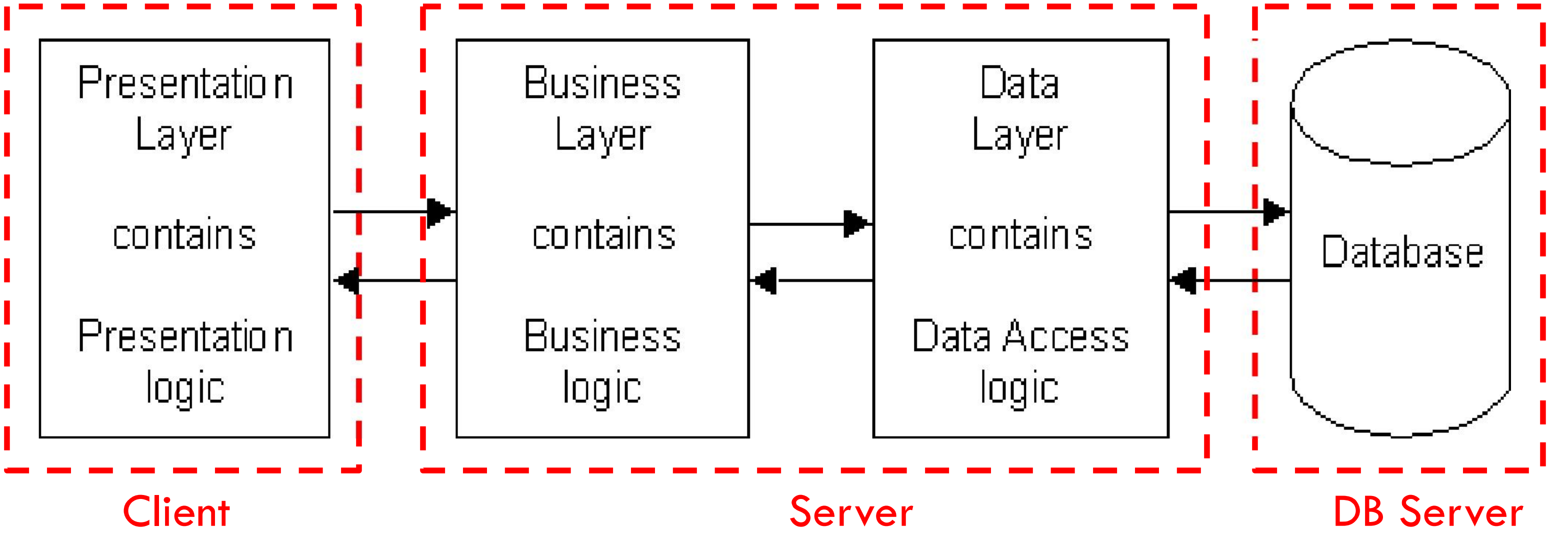
- All 3 layers are on the same machine
  - All code and processing kept on a single machine
- Presentation, Logic, Data layers are tightly connected
  - Scalability: Single processor means hard to increase volume of processing
  - Portability: Moving to a new machine may mean rewriting everything
  - Maintenance: Changing one layer requires changing other layers

2-Tier Architecture



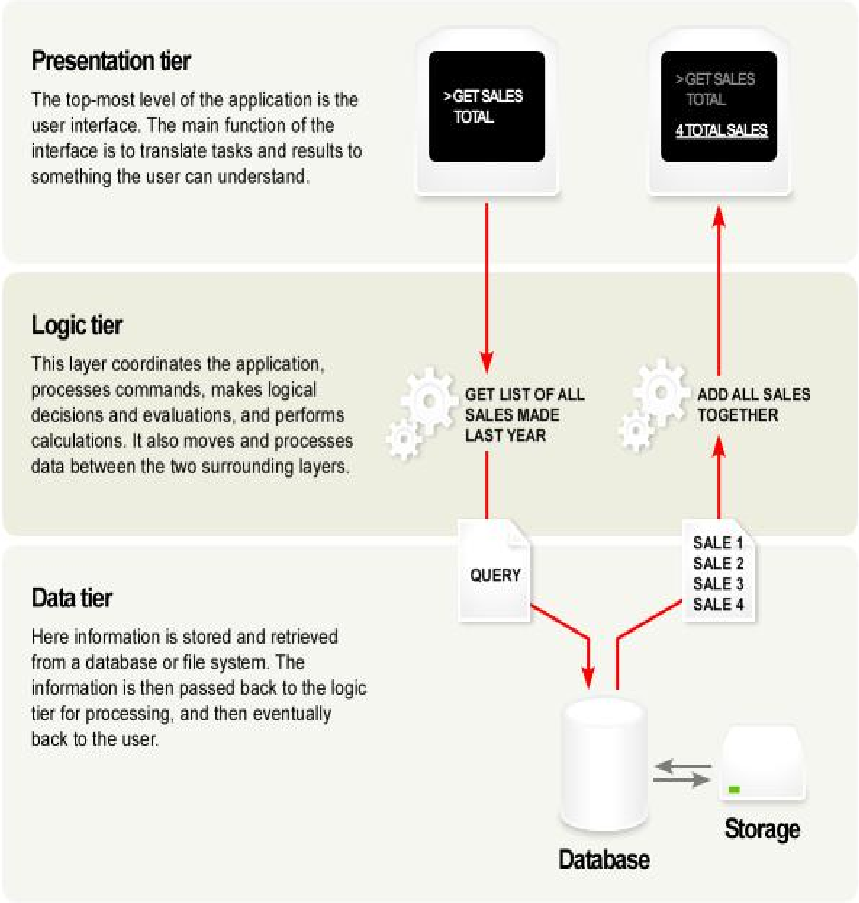
- Database runs on Server
  - Separated from client
  - Easy to switch to a different database
- Presentation and logic layers still tightly connected
  - Heavy load on server
  - Potential congestion on network
  - Presentation still tied to business logic

3-Tier Architecture



- Each layer can potentially run on a different machine
- Presentation, logic, data layers disconnected

A Typical 3-tier Architecture



Presentation tier

The top-most level of the application is the user interface. The main function of the interface is to translate tasks and results to something the user can understand.

Logic tier

This layer coordinates the application, processes commands, makes logical decisions and evaluations, and performs calculations. It also moves and processes data between the two surrounding layers.

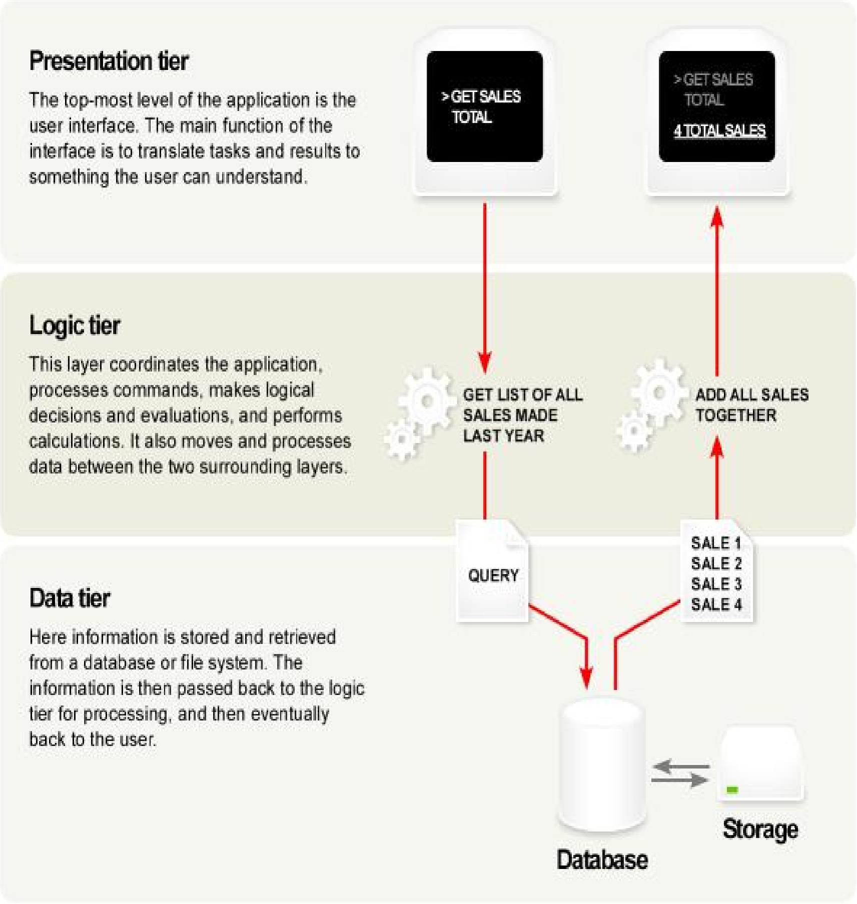
Data tier

Here information is stored and retrieved from a database or file system. The information is then passed back to the logic tier for processing, and then eventually back to the user.

Architecture Principles

- □ Client-server architecture
- □ Each tier (Presentation, Logic, Data) should be independent and should not expose dependencies related to the implementation
- □ Unconnected tiers should not communicate
- □ Change in platform affects only the layer running on that particular platform

A Typical 3-tier Architecture



Presentation tier

The top-most level of the application is the user interface. The main function of the interface is to translate tasks and results to something the user can understand.

Logic tier

This layer coordinates the application, processes commands, makes logical decisions and evaluations, and performs calculations. It also moves and processes data between the two surrounding layers.

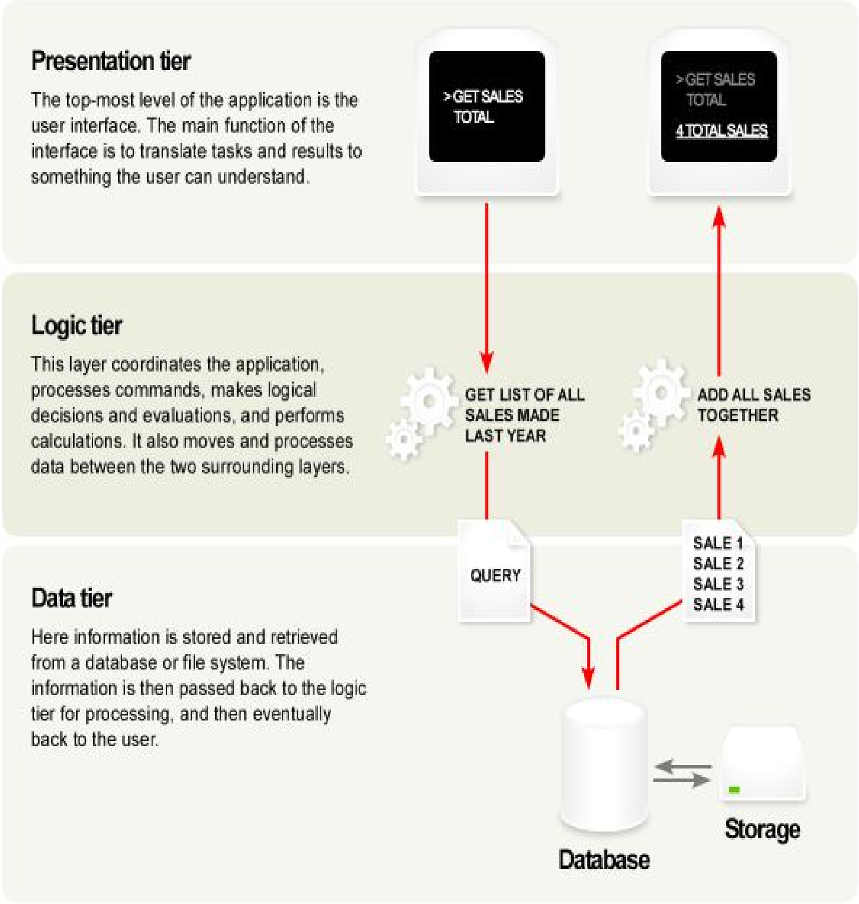
Data tier

Here information is stored and retrieved from a database or file system. The information is then passed back to the logic tier for processing, and then eventually back to the user.

Presentation Layer

- □ Provides user interface
- □ Handles the interaction with the user
- □ Sometimes called the GUI or client view or front-end
- □ Should not contain business logic or data access code

A Typical 3-tier Architecture



Presentation tier

The top-most level of the application is the user interface. The main function of the interface is to translate tasks and results to something the user can understand.

Logic tier

This layer coordinates the application, processes commands, makes logical decisions and evaluations, and performs calculations. It also moves and processes data between the two surrounding layers.

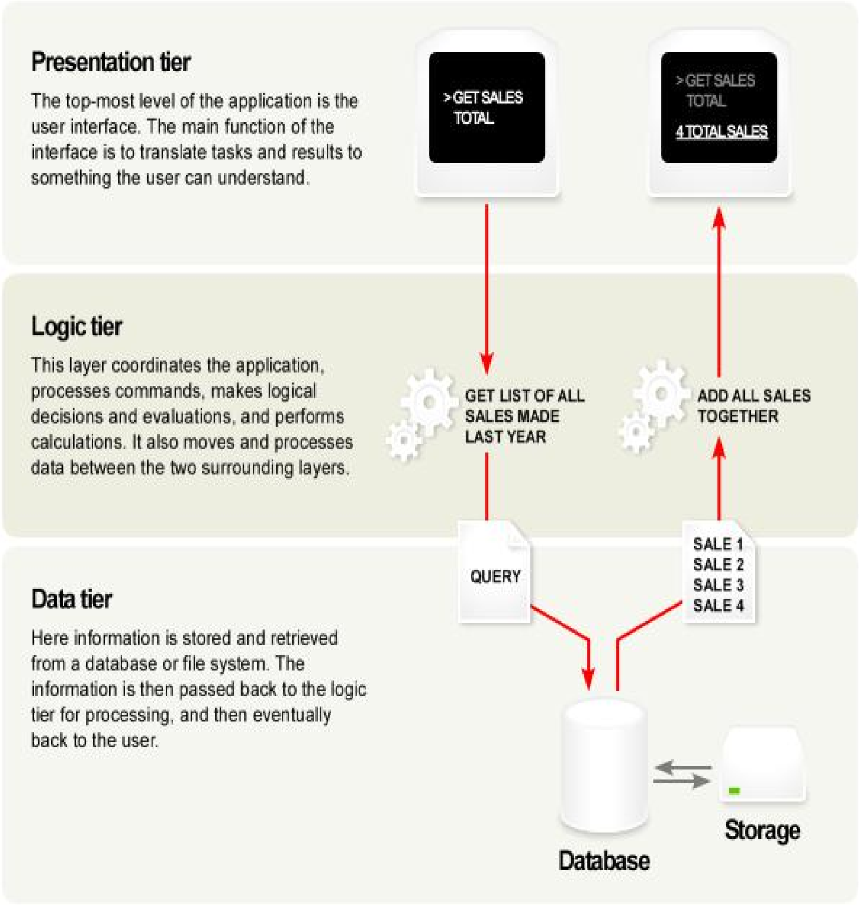
Data tier

Here information is stored and retrieved from a database or file system. The information is then passed back to the logic tier for processing, and then eventually back to the user.

Logic Layer

- □ The set of rules for processing information
- □ Can accommodate many users
- □ Sometimes called middleware/ back-end
- □ Should not contain presentation or data access code

A typical 3-tier Architecture



Presentation tier

The top-most level of the application is the user interface. The main function of the interface is to translate tasks and results to something the user can understand.

Logic tier

This layer coordinates the application, processes commands, makes logical decisions and evaluations, and performs calculations. It also moves and processes data between the two surrounding layers.

Data tier

Here information is stored and retrieved from a database or file system. The information is then passed back to the logic tier for processing, and then eventually back to the user.

Data Layer

□ The physical storage layer for data persistence

□ Manages access to DB or file system

□ Sometimes called back-end

□ Should not contain presentation or business logic code

The 3-Tier Architecture for Web Apps

- Presentation Layer

  Static or dynamically generated content rendered by the browser (front-end)

- Logic Layer

  A dynamic content processing and generation level application server, e.g., Java EE, ASP.NET, PHP, ColdFusion platform (middleware)

- Data Layer

  A database, comprising both data sets and the database management system or RDBMS software that manages and provides access to the data (back-end)

3-Tier Architecture - Advantages

- Independence of Layers
  - Easier to maintain
  - Components are reusable
  - Faster development (division of work)
    - Web designer does presentation
    - Software engineer does logic
    - DB admin does data model