

Dr Greg Wadley



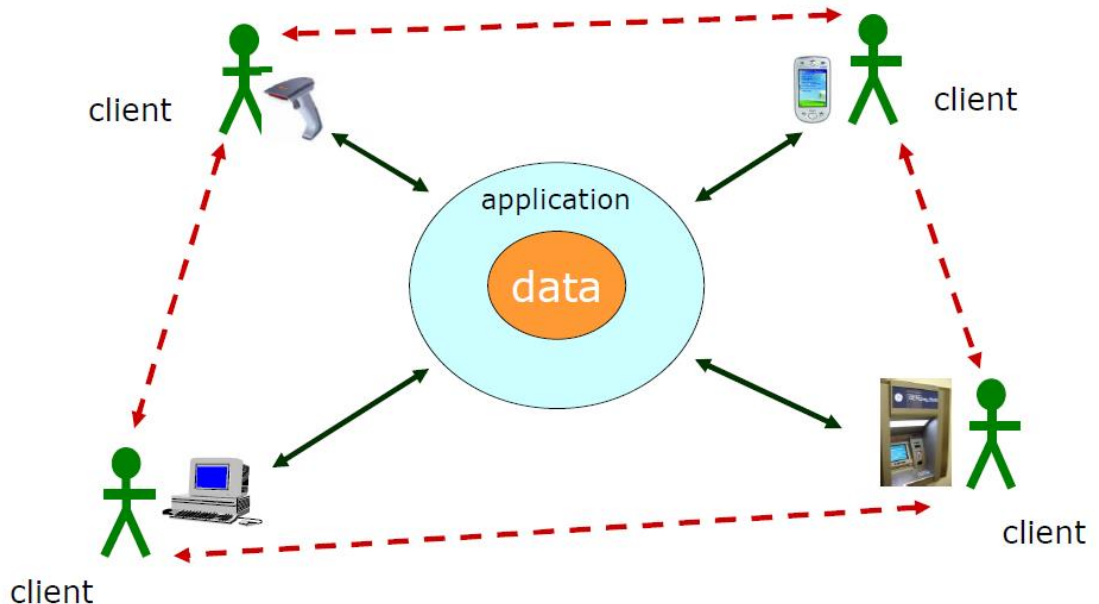
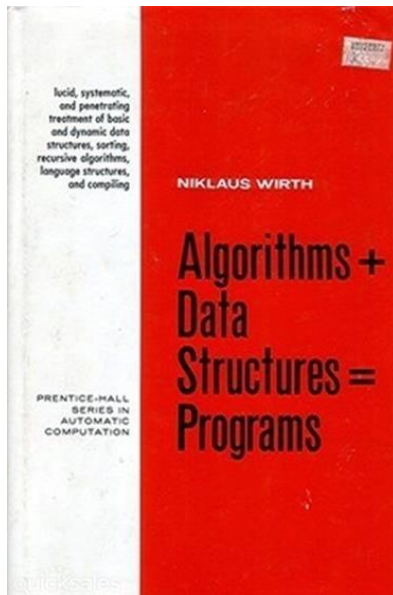
# INFO90002

## Database Systems & Information Modelling

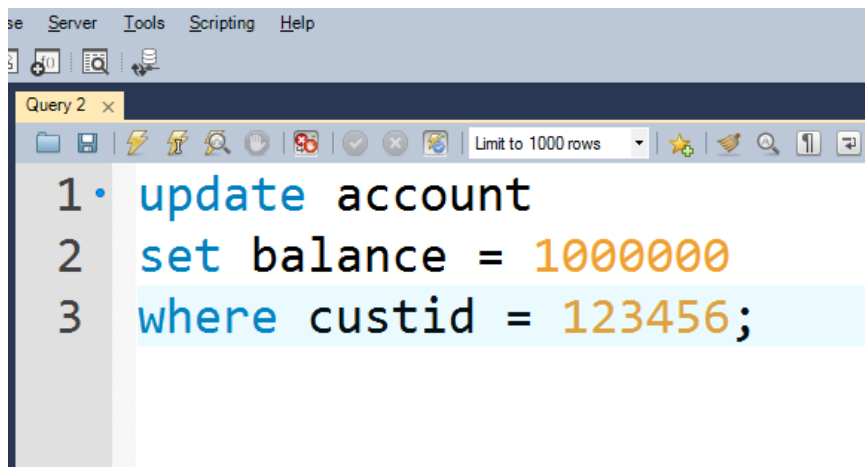
Week 07

Databases in Applications

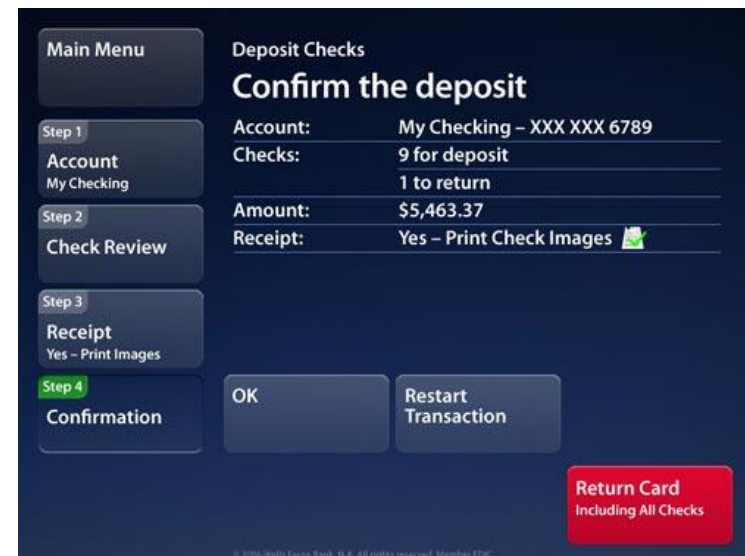
- How end-users access the database
- Business logic
- Stored procedures and triggers
- Embedding databases inside applications
- Application architectures



- SQL is declarative, intuitive, versatile, but ...
  - cannot express all possible queries in SQL
  - need to enforce business rules beyond domain/ref integrity
  - need procedural constructs such as loops and decisions
  - would you give end-users a query browser? Why not?
  - need a user interface that is both friendly and constraining



```
1 • update account
2 set balance = 1000000
3 where custid = 123456;
```



**Main Menu**

**Deposit Checks**

**Confirm the deposit**

**Step 1**  
Account: My Checking

**Step 2**  
Check Review

**Step 3**  
Receipt: Yes – Print Images

**Step 4**  
Confirmation

Account: My Checking – XXX XXX 6789  
Checks: 9 for deposit  
1 to return  
Amount: \$5,463.37  
Receipt: Yes – Print Check Images

OK Restart Transaction

Return Card  
Including All Checks



- Examples of business logic:
  - Check name and password. If good, login, if bad, error message
  - Insert one row in *Order* table, then several in *OrderItem* table
  - Check amount < balance. If so, subtract amount from one row in bank account table, then add amount to another row
  - For all rows in *Customer* table, send out monthly statements
- Procedural programming languages can do:
  - Sequence (several steps performed in order)
  - Iteration (loops)
  - Control flow (conditionals, decisions)
  - User interface (accept input and present output for users)
- SQL is specialized for low-level data access

- Customer places an order
  - Accept inputs from user (e.g. via web form)
  - Insert row into Order table
  - Repeat for each product ordered:
    - Check Product table shows sufficient quantity in stock. If so:
      - Insert one row into OrderItem table
      - Change Product table in-stock, Customer table amount-owing
  - If no errors encountered, end successfully
- Customer moves money from savings to credit card account
  - Accept inputs from user (via ATM, internet banking or mobile app)
  - Select balance from savings account
  - Is there enough money to withdraw? If so:
    - Update savings account balance = balance – withdrawal
    - Update credit card balance = balance + withdrawal
  - If no errors encountered, end successfully

- Need to combine data manipulation with the ability to handle sequence, iteration, decision. Different approaches:
  - “Embedded SQL”
    - “host language” = C, Fortran, Cobol, Java etc.
    - SQL statements are embedded in code and replaced with library calls during compilation
  - “Dynamic SQL”
    - host language sends SQL to DBMS via middleware e.g. ODBC
    - data is passed back to program as record-set
    - host language can handle business and presentation logic
    - example in next lecture “Web Applications”
  - Stored Procedures, Triggers
    - procedural code is stored and executed in the DBMS
    - enforce business logic within the database
    - in SQL-92 standard, but implemented differently in different DBMS

- Advantages
  - Compiled SQL statements
  - Faster code execution
  - Reduced network traffic
  - Improved security and data integrity
  - Business logic under control of DBA
  - Thinner clients
- Disadvantages
  - Code is not under the control of the application programmer
  - Proprietary language
    - e.g. MySQL SP's can't be used in Oracle or SQL Server

# Example stored procedure

1. accept person details as inputs
2. check whether the person is already in the database
3. if yes, return error
4. if no, add to database

(source: Hoffer chapter 8)

```
CREATE OR REPLACE PROCEDURE p_registerstudent
(
  p_first_name  IN VARCHAR2
  p_last_name   IN VARCHAR2
  p_email       IN VARCHAR2
  p_username    IN VARCHAR2
  p_password    IN VARCHAR2
  p_error       OUT VARCHAR2
)
IS
  l_user_exists NUMBER := 0;
  l_error       VARCHAR2(2000);

BEGIN
  BEGIN
    SELECT COUNT(*)
    INTO   l_user_exists
    FROM   users
    WHERE  username = p_username;

    EXCEPTION
    WHEN OTHERS THEN
      l_error := 'Error: Could not verify username';
    END;

    IF l_user_exists = 1 THEN
      l_error := 'Error: Username already exists!';
    ELSE
      BEGIN
        INSERT INTO users VALUES(p_first_name,p_last_name,p_email,p_username,p_password,SYSDATE);

        EXCEPTION
        WHEN OTHERS THEN
          l_error := 'Error: Could not insert user';
        END;
      END IF;

      p_error = l_error;
    END p_registerstudent;
```

Procedure p\_registerstudent accepts first and last name, email, username, and password as inputs and returns the error message(if any).

This query checks whether the username entered already exists in the database.

If the username already exists, an error message is created for the user.

If the username does not exist in the database, the data entered are inserted into the database.

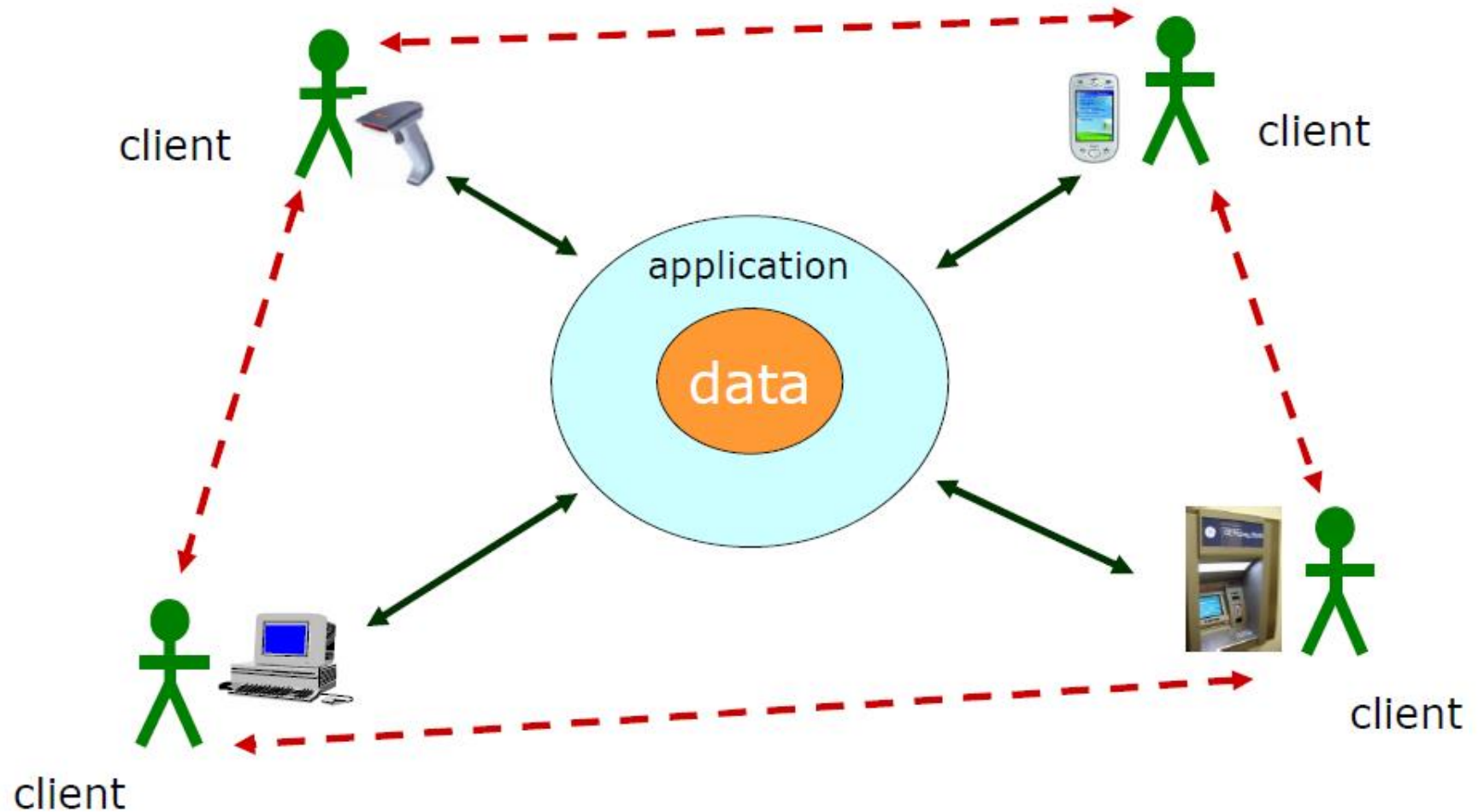




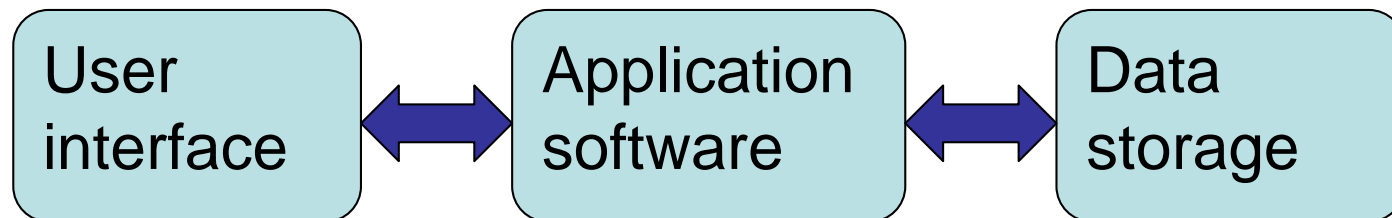
# Application Architectures

system architecture = “fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution”

~ ISO/IEC/IEEE 42010:2011  
*Systems and software engineering — Architecture description*



- An information system must provide
  - Presentation logic
    - input (keyboard, touchscreen, voice, sensor etc.)
    - output (large screen, printer, phone, ATM etc.)
  - Business logic
    - input and command handling
    - enforcement of business rules
  - Storage logic
    - persistent storage of data
    - enforcement of data integrity

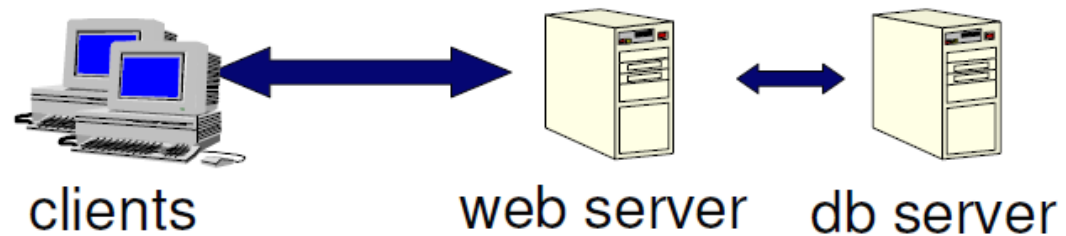


# Multi-tiered architectures

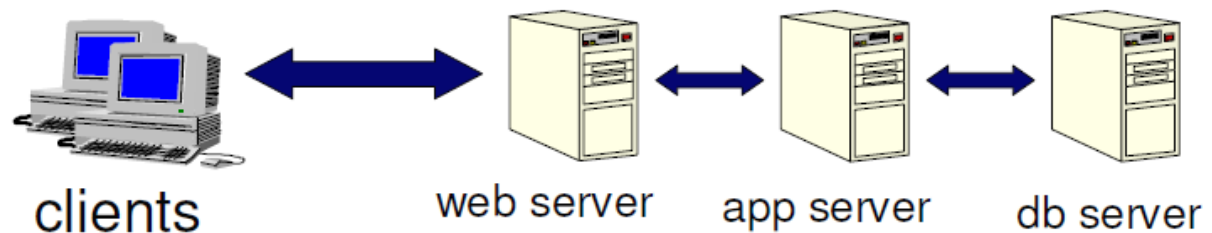
- 2 tiers



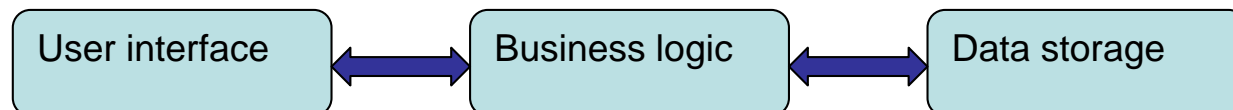
- 3 tiers



- 4 tiers



- Mainframe / dumb terminal
  - One large computer handles all logic
  - Problems: doesn't scale with number of users
- Client-Server architecture
  - 2-tier: e.g. file server, database, web
  - 3-tier: separation of Presentation, Processing and Storage logic
- Web architecture
  - a particular form of 3 or 4 tier architecture



# Mainframe (“1Tier”)

- Mainframes and mini-computers
- Dumb terminals (no processing at client end)
- Entire application ran on the same computer
  - Database
  - Business logic
  - User interface
- Enabling technologies included:
  - Embedded SQL
  - Report generators



- Server is a relational DBMS
  - data storage and access is done at the DBMS
- SQL queries sent to DB server, which returns raw data
- Presentation, business logic is handled in client application
- Platforms like Visual Basic (1990s into 2000s)

Bunga Raya Inventory System Ver 2.0 Copyright Liew Yoon Kiong

Brand: All Brands Category: All Categories Search View All

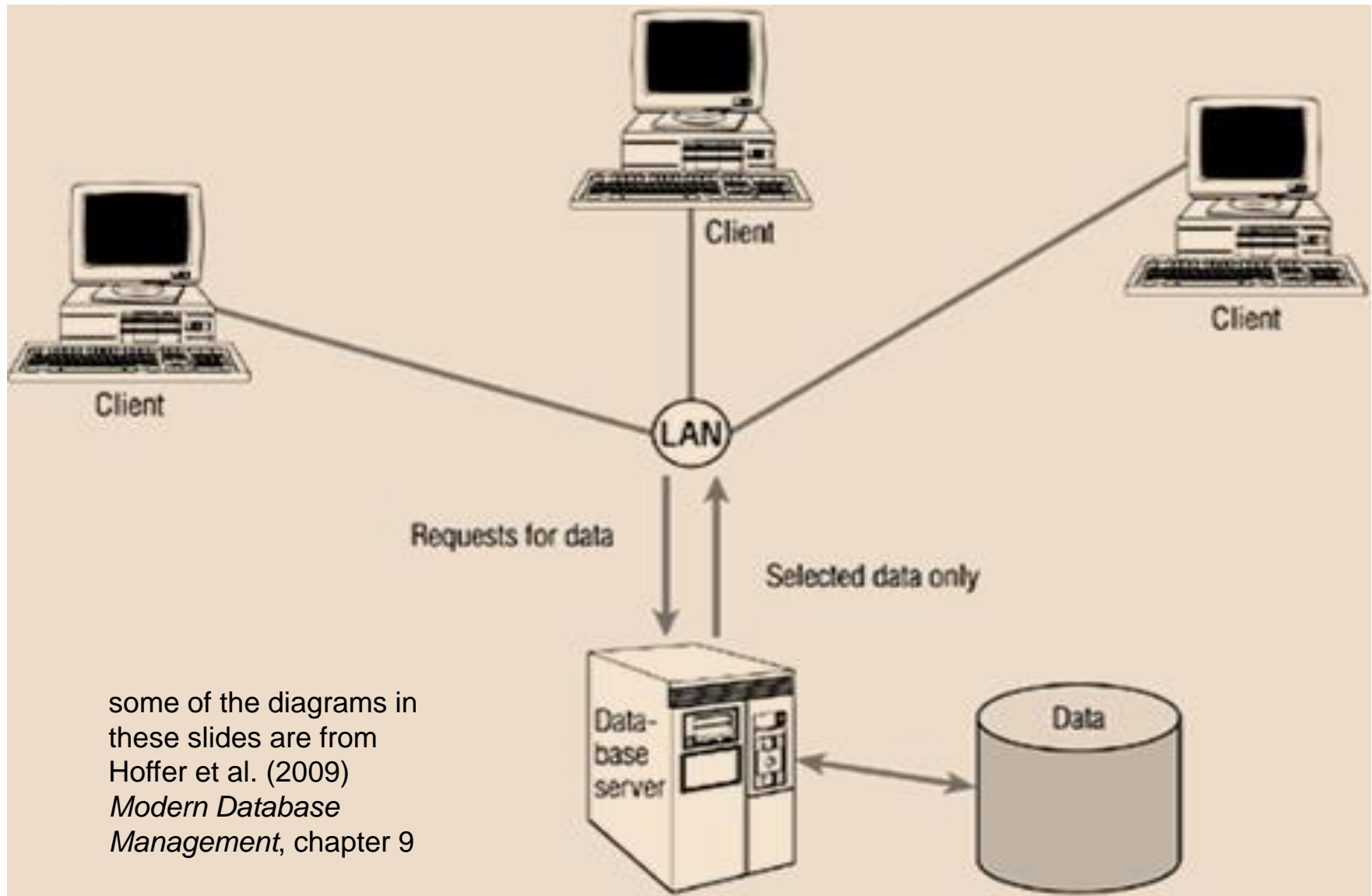
Category	Brand	Item Description	Serial Number	Stock
TV	Sharp	Sharp LED 60"	LC60LE630M	57
TV	Sharp	Sharp LED 32"	LC32LE240M	50
TV	Toshiba	Toshiba 29"		30
Smartphone	Samsung	Samsung Galaxy SIII		57
Smartphone	Motorola	Motorola Atrix2		15
Refrigerator	Sharp	Sharp Refrigerator	SJF72RVSL	25
Refrigerator	Sharp	Sharp Refrigerator	SJPT591	21
Refrigerator	Sharp	Sharp Refrigerator	SJPT491	30

Stock In and Out Record

Date	Category	Brand	Item Description	Serial Number	In	Out
28/01/2013	DVD Player	Haier	3D SoundTrack	H888	10	30
27/01/2013	Refrigerator	Sharp	Sharp Refrigerator	SJ151	20	5
27/01/2013	Oven	Sharp	Microwave Oven		20	10
28/01/2013	DVD Player	Sony	Blu-ray	SB1123	4	5
28/01/2013	DVD Player	Haier	3D SoundTrack	H888	10	4
28/01/2013	Washing Machine	Panasonic	Wash Machine	NA-F65B2	2	3
28/01/2013	DVD Player	Haier	3D SoundTrack	H888		
28/01/2013	Fan	Panasonic	Ceiling Fan			

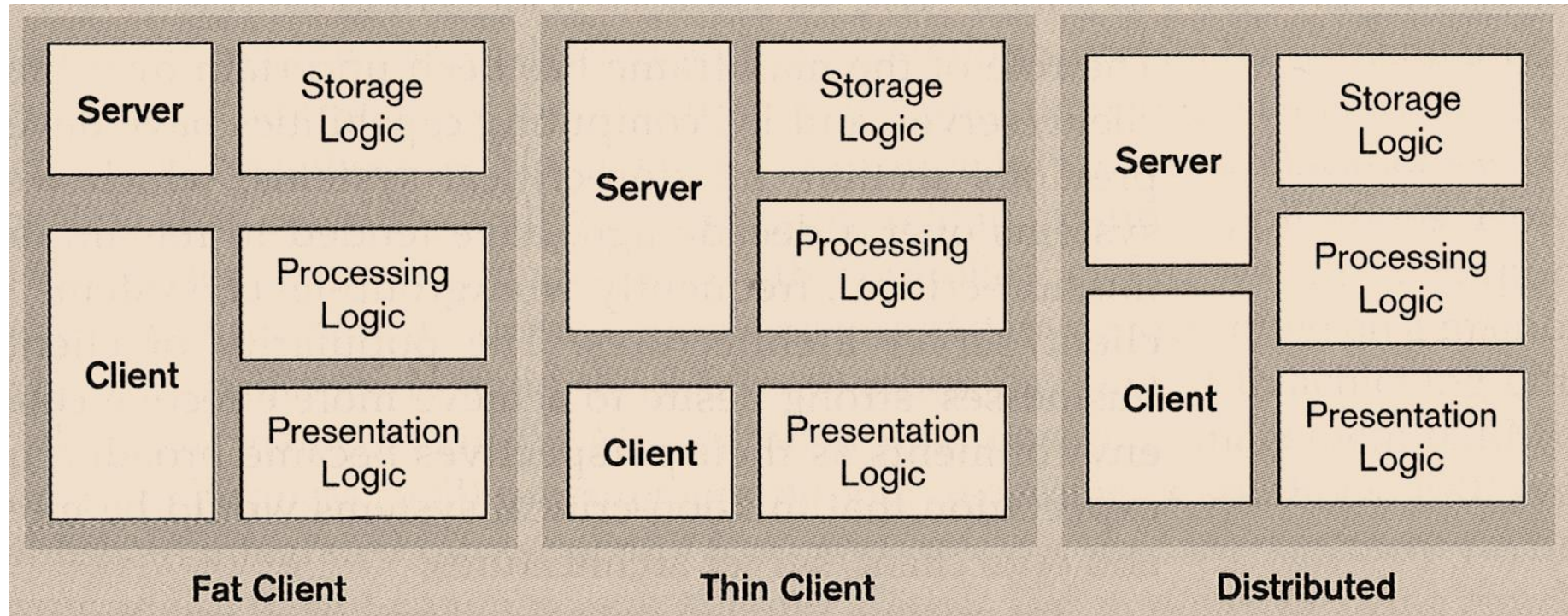
New Entry In Out Exit

# 2 Tier Example





- 2-tier distributions
  - Processing logic could be at client, server, or both

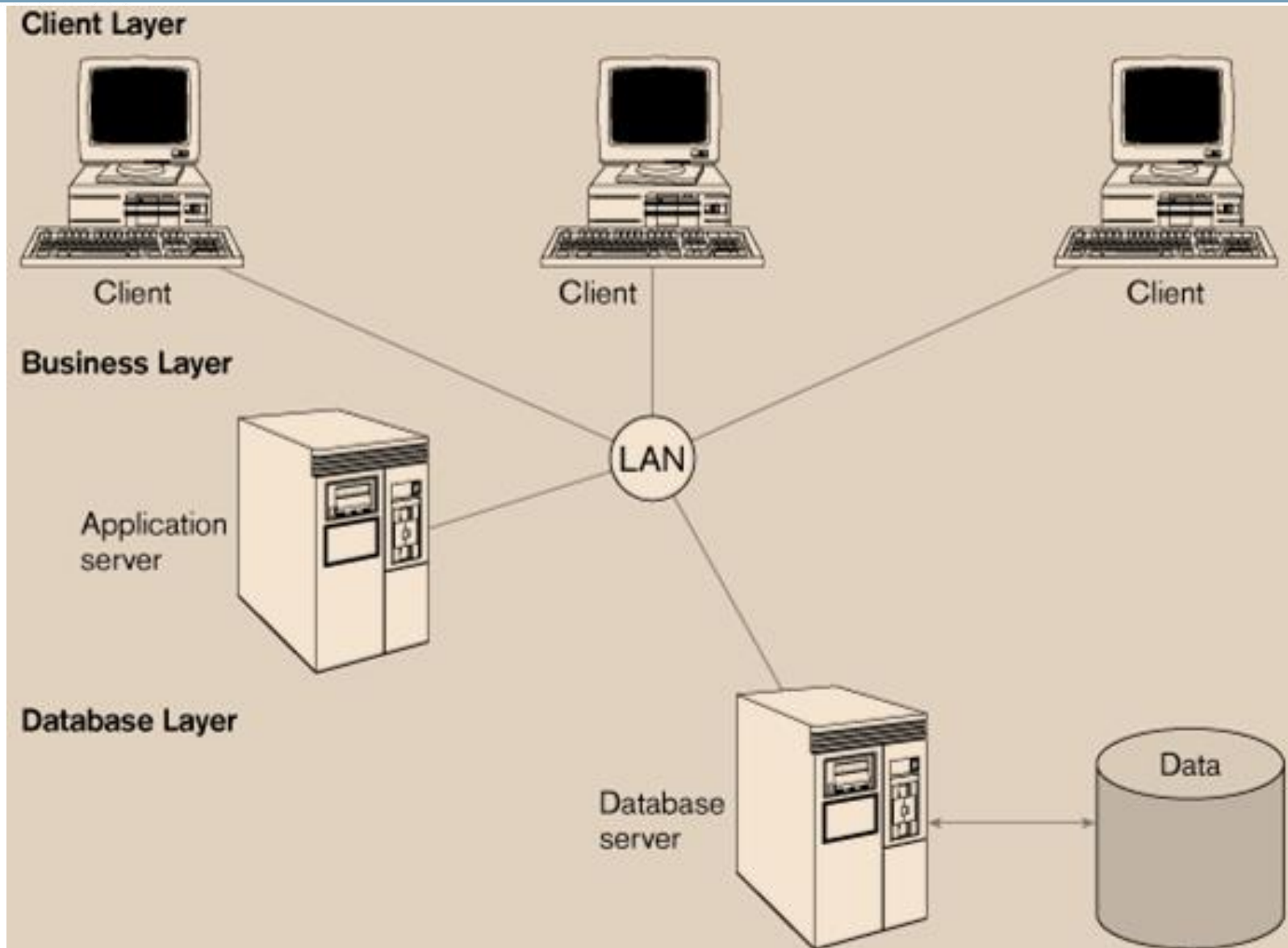


- Advantages
  - Clients and server share processing load
  - Good data integrity since data is all processed centrally
  - Stored procedures allow some business rules to be implemented on the database server
- Disadvantages
  - Presentation, data model, business logic are intertwined at client
  - If DB schema changes, all clients break
  - Updates need to be deployed to all clients
  - DB connection for every client, thus difficult to scale
  - Difficult to implement beyond the organization (to customers)

# 3-Tier architecture

- Client program <-> Application server <-> Database server
- Presentation logic
  - Client handles interface
    - Thinner clients
      - Limited or no data storage (possibly no hard disk)
- Business logic
  - Application Server deals with business logic
- Storage logic
  - Database server deals with data persistence and access

# A Three-tier architecture - Example



- Advantages
  - Scalability
  - Technological flexibility (can change business logic easily)
  - Can swap out any single component fairly easily
  - Long-term cost reduction
  - Improved security – customer machine does presentation only
- Disadvantages
  - High short-term costs
  - Tools and training
  - Complex to design
  - Variable standards

## 3-Tier (web based – see next lecture)

- Browser handles presentation logic
- Browser talks to web server via simple, standard protocol
- Business logic and data storage handled on server(s)
- Pros
  - Everyone has a browser
  - No need for install and maintain client software
  - HTML and HTTP are simple standards, widely supported
  - Opens up the possibility of global access to database
- Cons
  - Even more complexity in the middle-tier
  - Simple standards = hard to make complex application
  - Global access = potential security nightmare (next page)

- Network environment creates complex security issues
- Security can be enforced at different tiers:
  - application password security
    - for allowing access to the application software
  - database-level password security
    - for determining access privileges to tables
  - secure client/server communication
    - via encryption



## Create an account

It's free and always will be.

First name  Surname

Email or mobile number

Re-enter email or mobile number

New password

Birthday

Day  Month  Year  Why do I need to provide my date of birth?

☐ Female ☐ Male

Personal customers Business customers [Help ?](#)

Enter your customer ID (Using your keyboard)

Enter your password (Using the buttons below)

1	2	3	4	5	6	7	8	9	0			
A	B	C	D	E	F	G	H	I	J	K	L	M
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

[Forgotten your password?](#)

Start Over University Login Guest Login Discovery BONUS+ Interlibrary Loans Search Other Libraries Program Calendar

(Search History)

KEYWORD  SQL Search Entire Collection

☐ Limit search to items available for borrowing or consultation  
438 results found. Sorted by **relevance** | [date](#) | [title](#)

Result page: 1 [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) ... [37](#) [Next](#)

[Save Marked Records](#) [Save All Records](#) [Save Marked Records to List](#)

KEYWORDS (1-12 of 438)		
SQL found in main title of entries 1-195		
1	<a href="#">Beginning Oracle SQL : for Oracle Database 12c / Lex De Haan, Tim Gorman, Inger J#nsen, Melanie Caffrey.</a> Berkeley : Apress, Third edition. 1 online resource.	2014.
2	<a href="#">Oracle PL/SQL programming [electronic resource] / Steven Feuerstein, Bill Pribyl.</a> Sebastopol, Calif. : O'Reilly Media, 6th ed. 1 online resource (1 v.) : ill.	c2014.

**BONUS+** [Discovery](#) [Trove](#) [CARM](#)

Sign in to continue to Gmail

☒ Stay signed in [Need help?](#)