DOMAIN 8

CH: 20 SOFTWARE DEVELOPMENT SECURITY

Custom s/w development to achieve unique business objectives

→ security Vulnerabilities

Malicious code

Backdoors

Attackers

... Rely on 3rd party code & libraries

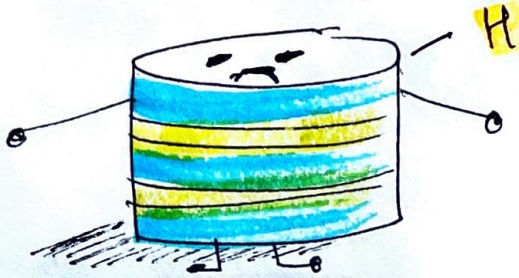
PERSPECTIVE →

Inject Security control into SDLC - S/W Development life cycle.

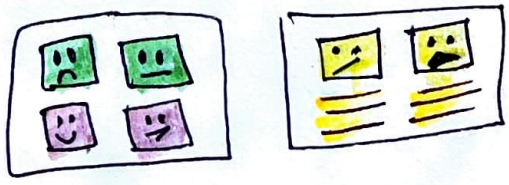
BE AWARE - WHOM WE NEED PROTECTION FROM?

Malicious code

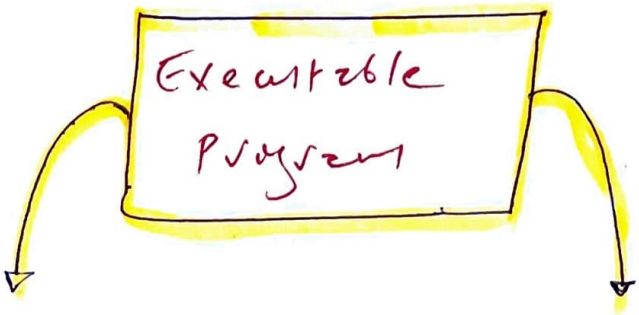
Database Vulnerabilities



Application Development Flaws



* S/W Development



Compile

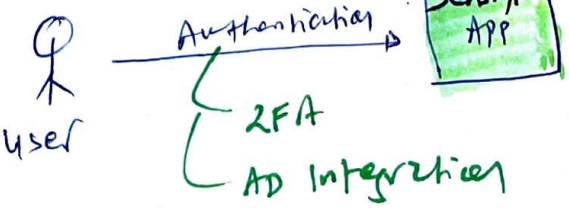
- Code convert into executable file then distribute to end users — not the source code
- * Programmer can embed malicious code
- C, JAVA

Interprete

- Distribute source code to end users
- * User can review code & insert malicious one
- Python, Javascript

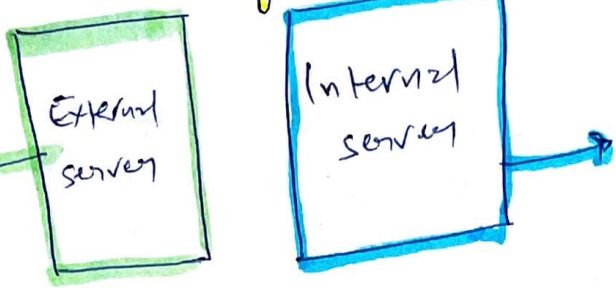
Avoiding & mitigating system failure with some methods.

1. Input validation (from user) ← occurs on server side of transaction
   - ⇓
   - LIMIT CHECK + ESCAPING INPUT
   - DoB <integer>
   - state <three word VIC>
2. Authentication & Session mgmt
   - Session tokens



Level of Authentication ∝ sensitivity of application

3 ↳ Error Handling



4 ↳ Logging = SIEM = stack driver

5 ↳ Fail-secure & Fail-open = 2 choices of planning system failure

- - - Famous BSOD

This is better for security reasons

Enable full security or disable all to diagnose problem to restore system

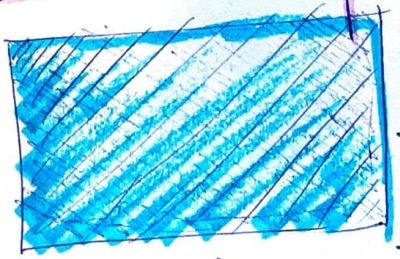
Bypass security controls.

BLUE SCREEN

(Fail-secure response)

Blue screen of Death (BSOD)

= STOP Error



- Application gaining direct access to H/W
- Attempt to bypass security check
- Memory interference b/w two process

often we → Disable security in s/w for easy installation

* Software Development life cycle (SDLC)

7 Core activities for secure system Development:

1. Conceptual Definition
   - project purpose & system requirements
2. Functional Requirement Determination
   - Input
   - Behavior
   - Output
   - } For all system requirements
3. Control specification Development
   - Security controls
     - Access controls for authorised users
     - maintain confidentiality & data using Encryption + Data protection methods
     - Address Availability & Fault-tolerance
     - Enforce Audit + Accountability or detective mechanisms for illegal activity.
4. Design Review
   - Based on 3 & 2
5. Code Review walk-through
   - check for problems in logic flow / design / security flaw.
6. User Acceptance Testing (UAT)
   - All sort of user testing before code moves to deployment / prod.
7. Maintenance & change mgmt.
   - Deploy

* lifecycle models

Gantt chart & PERT

S/W capability maturity model

(SW-CMM) idea = quality of s/w depends on quality of development process.

- 5 stages (P.T.O)

Agile model

- Supports Devops with continuous delivery + deployment

IDEAL model

- Implements many SW-CMM attributes

Spiral model

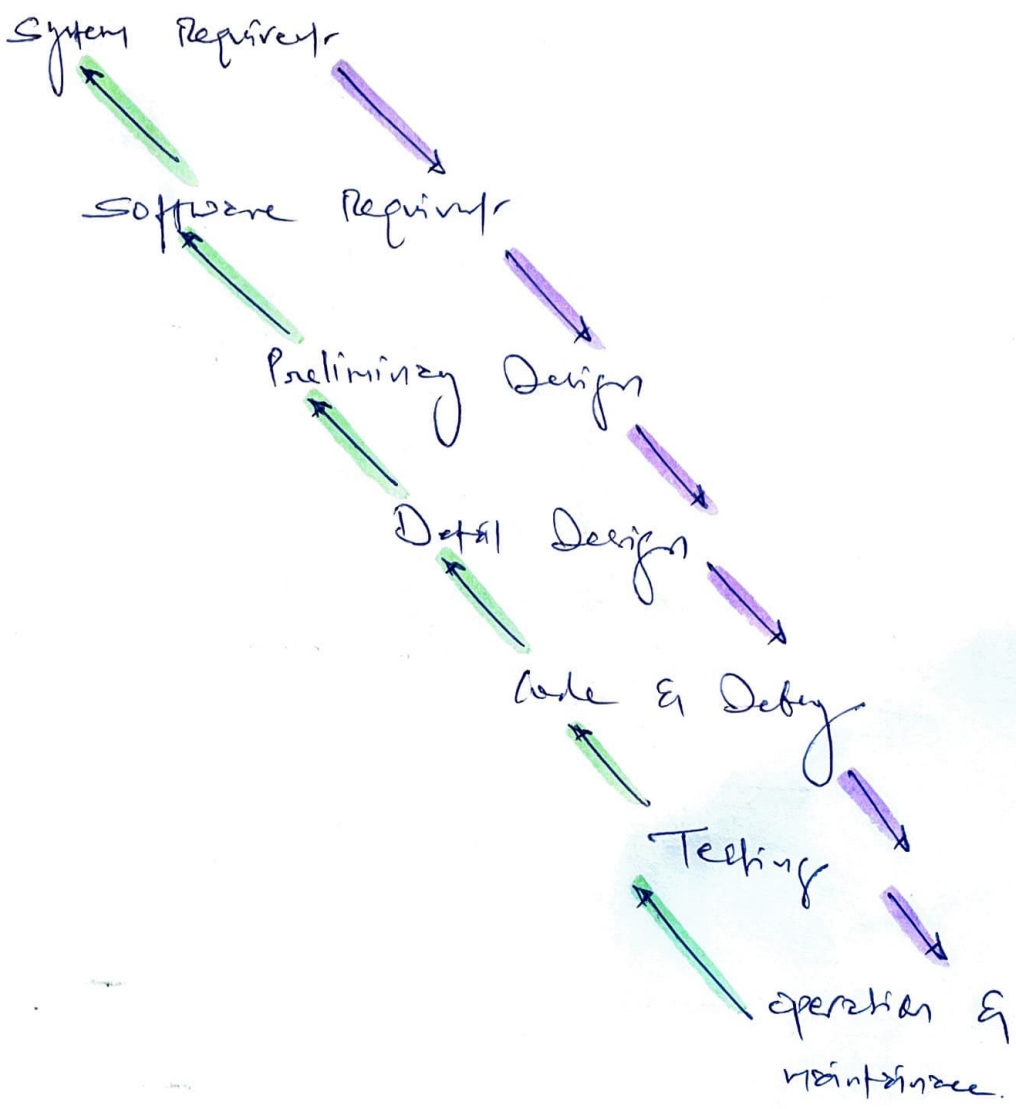
- Multiple iterations of waterfall model
- Each loop of spiral results in the development of new system

Waterfall model

- System development lifecycle as a series of iterative activities.

↳ Developer can sit back & watch till five vises in production.

Waterfall Model



IDEAL Model

1. Initiating – Business reasons to change are outlined
2. Diagnosing – Analysis & current state of organization + recommendations
3. Establishing – Develope specific plan based on recommendation & change
4. Acting – Stop "talking the talk" start "walk the walk"
5. Learning – Feedback → continuous improvement

Software Capability Maturity Model (SW-CMM)

- Quality of s/w & Quality of development process.

5 stages.

Level 1 : Initial

- Little to no defined s/w development process

Level 2 : Repeatable

- Basic lifecycle mgmt process are introduced such as reuse of code.

Level 3 : Defined

- Developer operate according to set of formal, documented s/w development process.

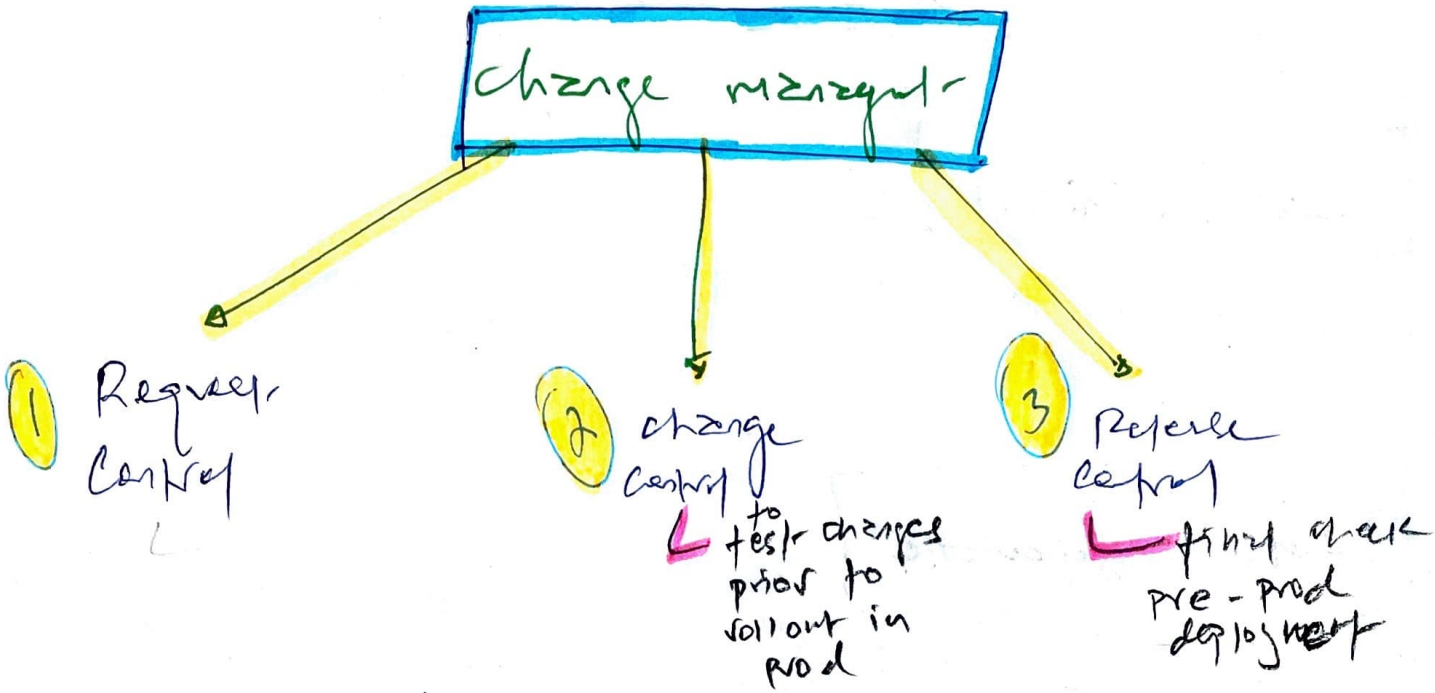
Level 4 : Managed

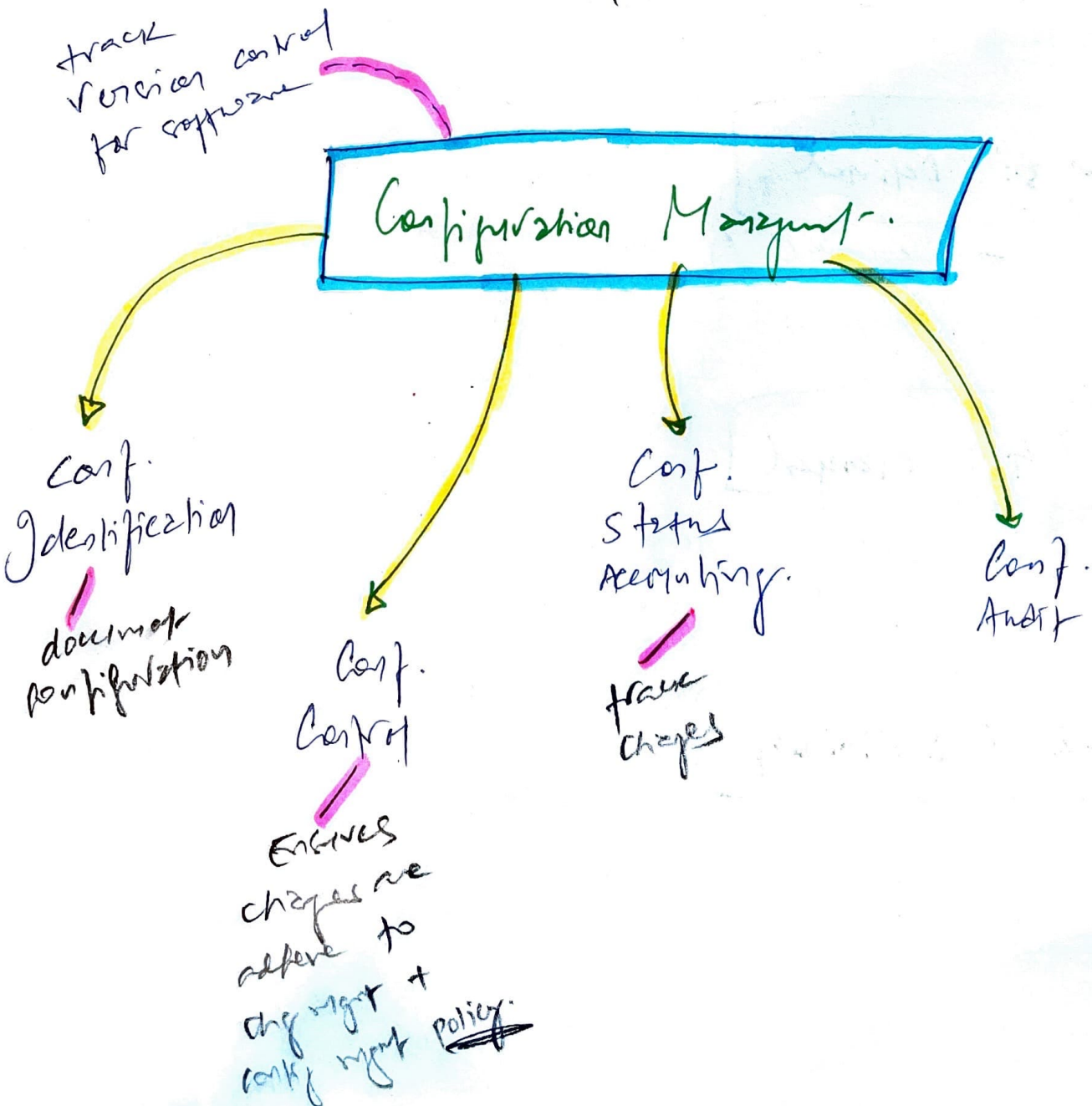
- Quantitive measures are utilized to gain detailed understanding of development process.

Level 5 : Optimizing

- process for continuous improvement

* change & Configuration Mangemt.

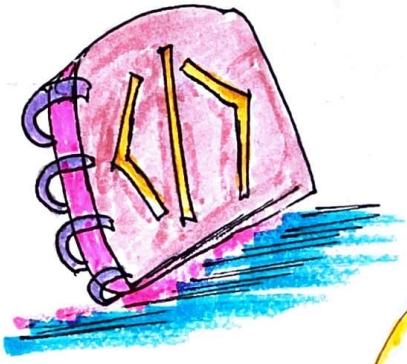


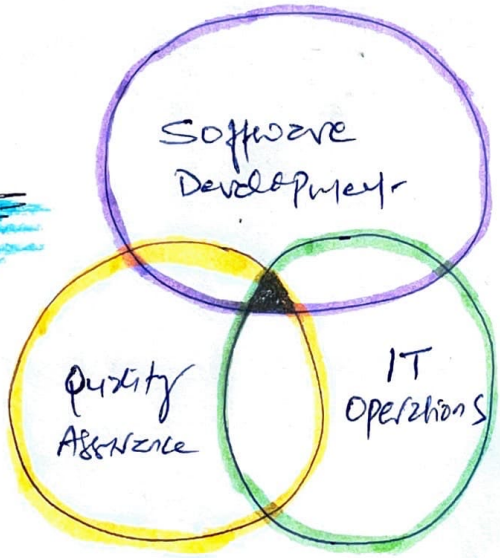


* DevOps Approach --- Aligned --- Agile Development Approach

Create code
Test code
Deploy code

→ 3 function into one operational model





Information security is NOT part of DevOps.

API

API Keys = To identify Project / Application (nothing to do with user)

Restrict your API key

Don't put it in public repo

Project Identification

project authentication

User → Authenticate

→ Authorize

Software Testing

Perform a test called Reasonable check

Note - live workload is best stress testing but not advisable in early development as it can violate C & I

3 Methods

i) White-box testing
- test internal code + logic + structure

ii) Black-box testing (E.g Fuzz Testing)
- From user perspective (input + output scenarios)
- they don't have access to internal code

iii) Gray-box testing
- Hybrid : popular for s/w validation

Application Security

Static testing (SAST)
- For pre-prod / source code
- Without running app. or source code

Dynamic testing (DAST)
- For runtime / prod
- Web Application Testing: E.g SQL injection for web app + XSS

Software Acquisition

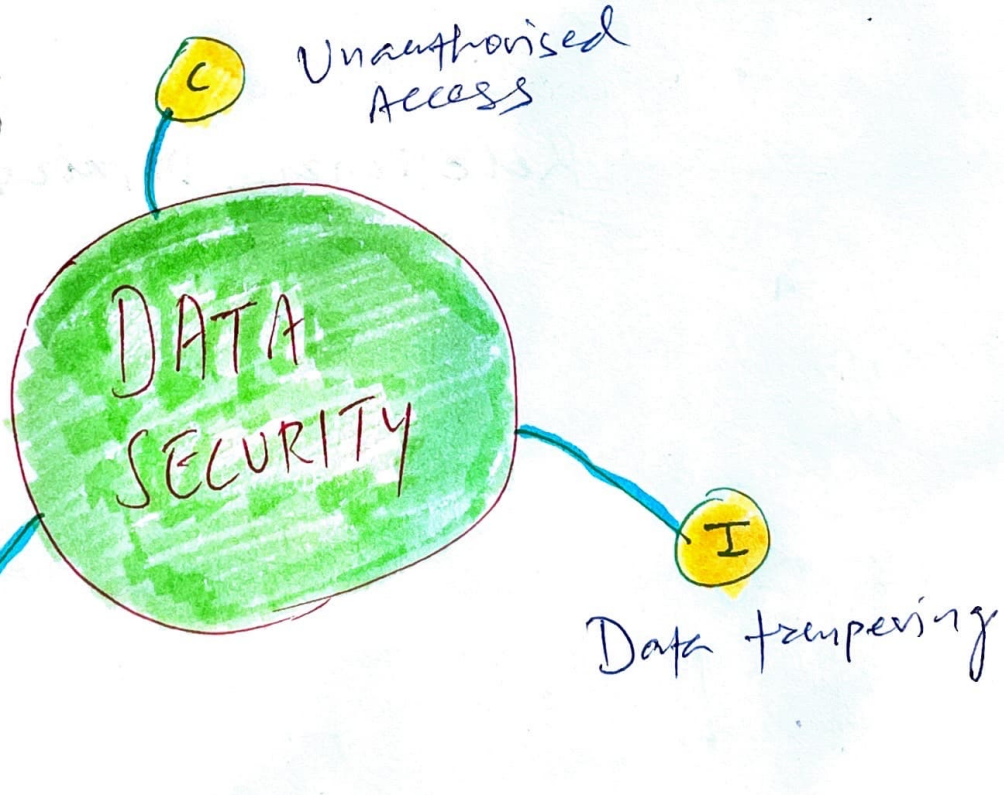
I22S

- Physical / virtual Exchange server
* on-house security

S22S

- Gmail
* Monitor vendor's security

DATABASE & DATA WAREHOUSING



Database Architecture

1-M Hierarchical Database

- organization chart
- one database
- one to many relationship.

M-2-M Distributed Database

- data stored in more than one database but they are logically connected.
- may-to-may relationship

cardinality = Rows
Degree = Columns

Relational Database

Candidate Keys

ID	Email	mobile

Primary Keys

selected as primary candidate → ID

Foreign Keys

Referential Integrity → mobile

Database Table → compliance with normal Form → Normalization (pt. of 2.)

- 1NF
- 2NF
- 3NF

Database language → SQL

SQL → 2 components

DDL - Data Definition Language

- Allows creation & modification of database structure (schema)

DML - Data Manipulation Language.

- Allows users to interact with data contained within that schema.

Database Transactions

Implicit and Explicit use of transaction → Ensures → DATA Integrity

*Adding & removing fund executed as one transaction, not seperate.

Database transactions 4 characteristics

ACID

Atomicity

transaction = All or nothing

(A) Durability

Once data is committed to database = it must be **Preserved**

Using Backups, transaction logs

(I) Consistency

All transaction use = Consistent / updated data

Used to organize data in relational database.

(I) Isolation

logistic inventory goods

Same Data

transaction 1 — then only — transaction 2

Database Normalization — Process of bringing database into compliance with various forms

Have to go in this order ↑

3NF
↑
2NF
↑
1NF

These forms adds requirements to reduce redundancy in tables, eliminating misplaced data in tables, & perform other housekeeping tasks.

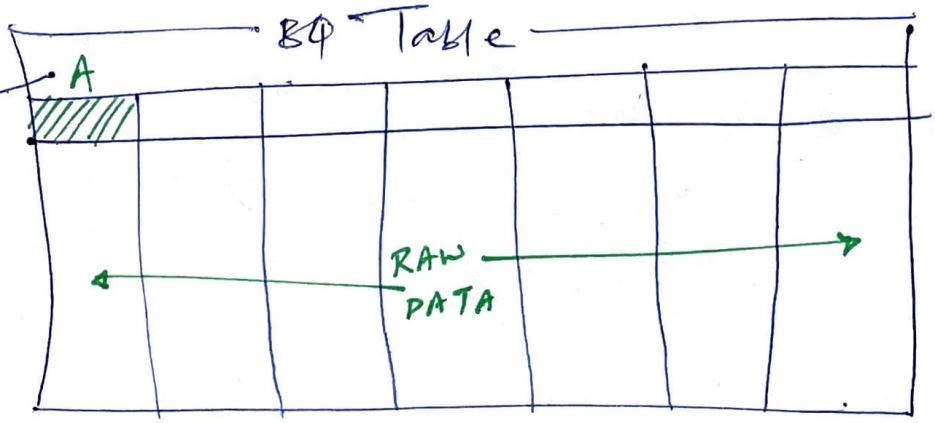
move DOWN tables

DATABASE SECURITY

(From DB Admin perspective)

→ Data classification as a go-to-start security

BUT



only access to column A rather than whole table

=

DATABASE VIEW

1 ---- multilevel security in Database

- Views are stored as SQL commands rather than tables of data. — or query

(Edit control) CONCURRENCY

2 ---- Preventive security mechanism to protect Integrity & Availability.

- Two different processes update same database and unaware about each others activity.

without concurrency

- Lost updates
- Dirty Reads

Lock feature (Similar to Palo Alto's COMMIT)

- Process reads a record from transaction that successfully did not commit

(SQL)

When multiple security required for database, it's better to keep requirement separate. Mixing classification levels / need-to-know requirement is known as Database Contamination

Semantic Integrity

3 - To ensure user's action don't violet structural rules.

- Security feature for DBMS
- Checks all stored data types are within valid domain range, ensures the logical values exists

Employ time & date stamps

4 To maintain data integrity & availability.

Granular control (Using OBJECTS)

6 5

Content-dependent access

Based on content of payload or when object being accessed

relates

Context-dependent access control

How each object / field relate to overall activity / communication

Cell suppression

Hide individual database field / cell.

Polyinstantiation

(Prevents unauthorised user to view classified info)

P.I. end

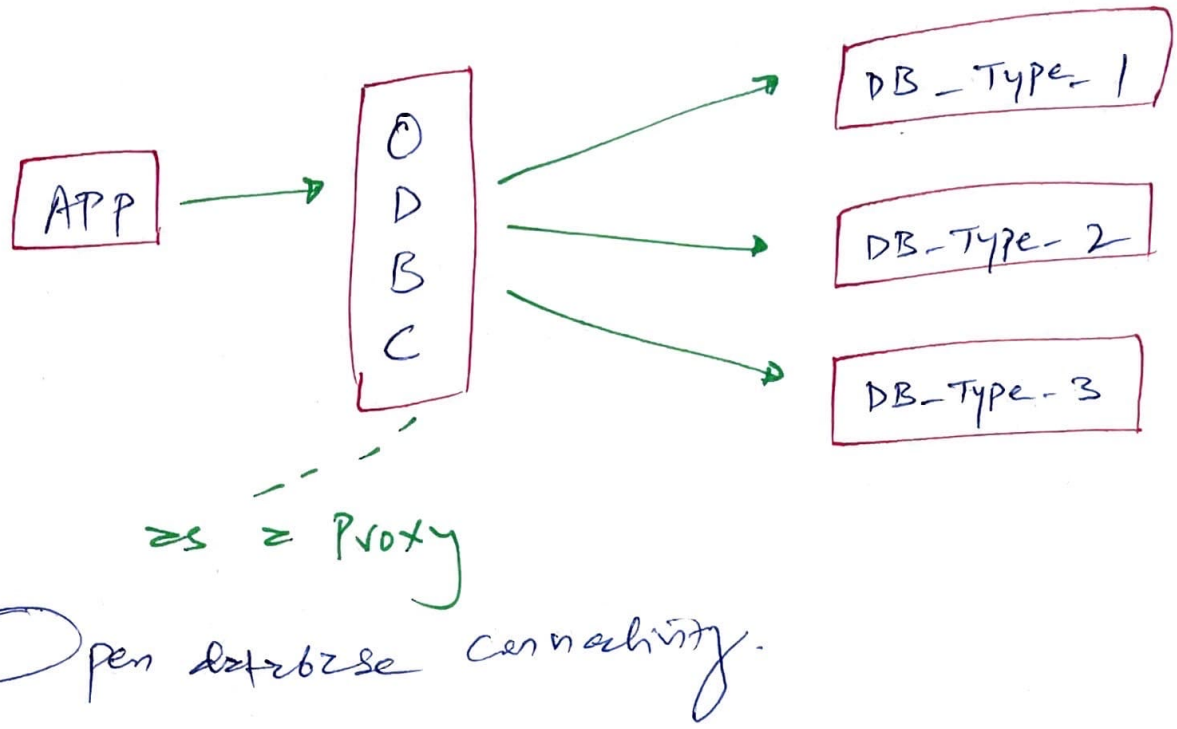
7

- Two rows, same primary keys, different classification levels.

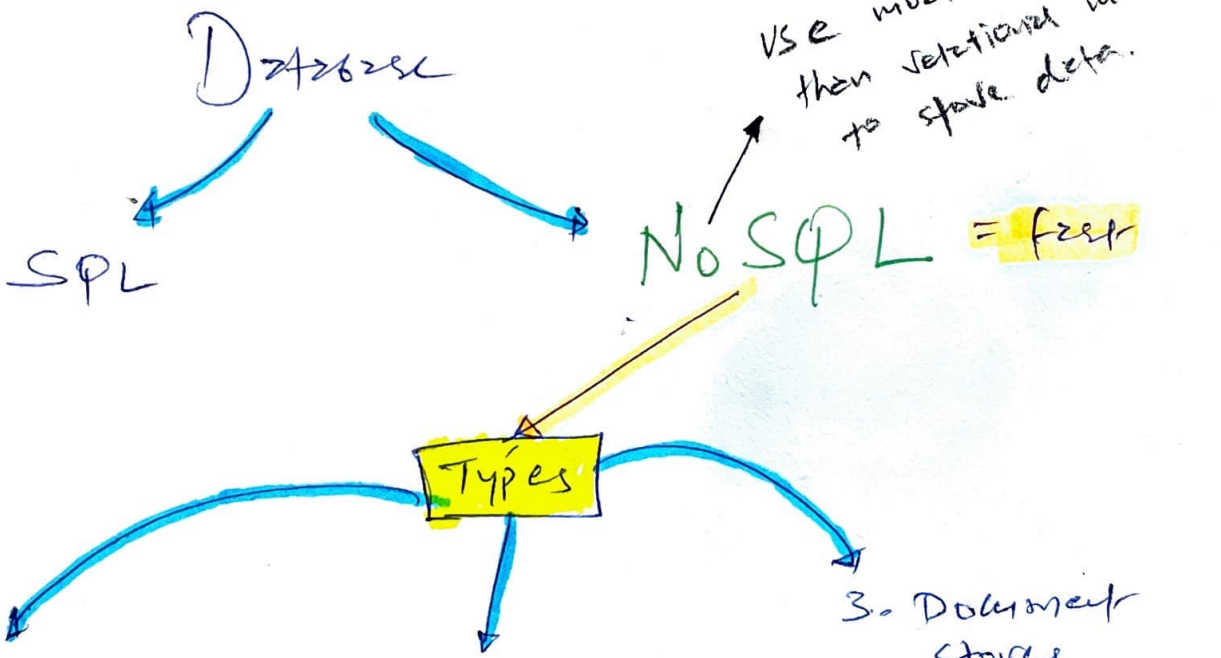
Ship46 ---- top secret = undercover

1 2 3 ---- secret = normal routine

Other ships



Open database connectivity.



1. Key/value stores

1- Dave
2. Kyuuzi
3. Rocks
4. Always

2. Graph Database

3. Document stores.

- XML
- JSON

Data Storage

Not just security of data, consider types of storage too ---

1. Primary storage = RAM = volatile / Real
   - CPU
2. Secondary storage = DVD / USB (Non volatile)
3. Virtual Memory = simulate primary storage
4. Virtual storage = simulate secondary storage
   - E.g RAM DISK
5. Random Access = as can request content from any point within media
   Storage
   - RAM, Harddrive
6. Sequential Access storage = requires scanning through entire media to reach specific address
   - Magnetic tape
7. Volatile storage = RAM (system CPU)
   - Power-off = data gone
8. Non-volatile storage = DVD / USB
   - Power-off doesn't matter

How insecure is data in Database!

Storage threats

Unauthorised Access

(C) Confidentiality

Controls

- Encrypt data
- Store key in secure vault
- Least privilege / RBAC
- Monitoring
- classification of data + Asset
- ★ Ensure data on higher classification level is not readable at lower classification level.

Covert channel Attacks

Transmission of sensitive data b/w classification levels, through direct / indirect manipulation of shared media.

SPECTRE & MELTDOWN

* Knowledge Based Systems - AI

Expert Systems

= No Emotions for investment decisions.

① Knowledge Base
- Expert human knowledge codify into series of if / then statements

② Inference Engine
- Analyse ① Knowledge base & advise or appropriate decision.

10 Thriller Films You might like

watch "Pi"

Extension → Neural Network

Neural Networks

- Attempt to imitate biological logical reasoning process of human mind.
- Extension of machine learning / deep learning / cognitive system

= Superior than AI

- Delta learning rule: neural networks are able to learn from experience.

(DEEP) Machine Learning

Extension → Neural Network

Model to detect malicious logins

Supervised Learning (labeled data)

model Trains → prediction

Unsupervised Learning (Unlabled data)

independent model based on their own learning.

Analyst learn malicious login attempts & build a model

REMEMBER

IDEAL model & SW-CMM model

"I ... I — Dr. Ed ... Am Low"

IDEAL	SW-CMM
Initial	Initial
Diagnosing	Repeatable
Establishing	Defined
Acting	Managed
Learning	Optimized

SDLC

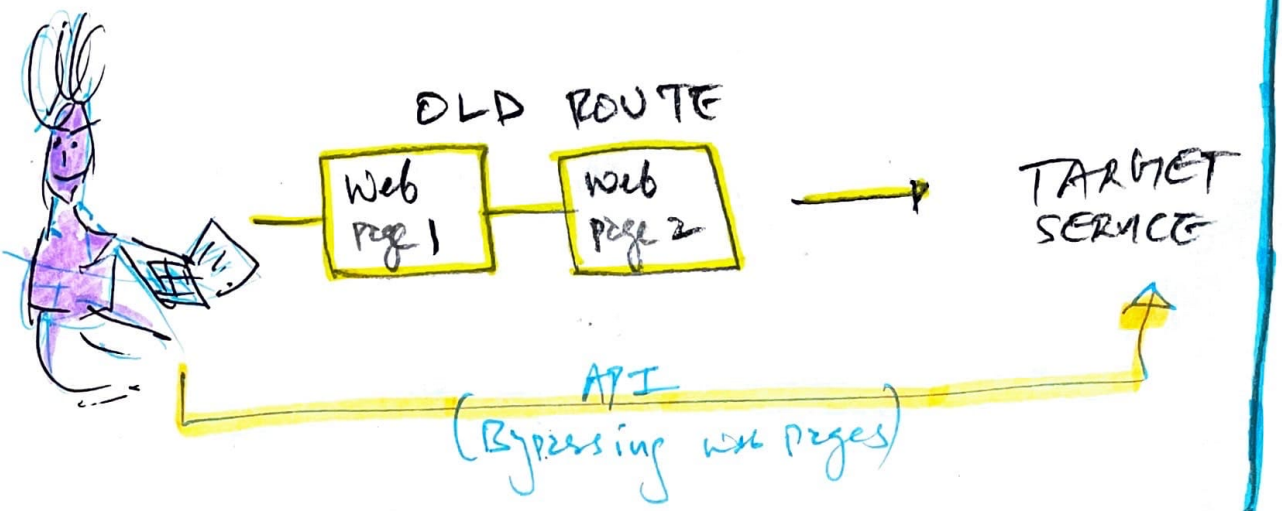
C F C D C S M

- Conceptual definition
- Function req. — Input, Behavior, Output
- Control specification development
- Design Review
- Coding Review
- System test Review
- Maintenance & chg. mgmt.

Waterfall

SR^2 PD DD C T O

- System requirement
- Software requirement
- Preliminary Design
- Detailed Design
- Code & Debug
- Testing
- Operation & Maintenance



Polyinstantiation

- 2 or more rows have same primary keys but contain different data for use at different classification levels.
- Polyinstantiation is used as a defence against INFERENCE ATTACKS.

Noise & Perturbation Concept

Admin deliberately adds false / misleading data into DBMS to redirect or thwart INFORMATION CONFIDENTIALITY ATTACKS.