NETWORK(NW): A group of devices that are connected together to communicate and share resources

+: Shared resources; Easy access; Collaboration

-: High cost; Security breach; Failure; Maintenance.

WAN: NW that spans large area, even across countries

LAN: NW that is self-contained, spans small area

INTERNET: Largest global WAN, public owned, contains a large number of Intranets

INTRANET: Private NW that uses protocol & services to share company info. Can be accessed from Internet.

+: Security, privacy; speed; cost

SWITCHING: Tech$^{\underline{n}}$ to transmit data over NW to devices

CIRCUIT ~: Dedicated channel est. before data transmit

PACKET ~: Data broken into small units (packets), each packet takes best route avail and travels independently

+: Efficient; More secure (multiple route hard to attack)

-: Sequence altered; requires assembly at destination

MODELS: Explain data comm processing in soft/hardware; blueprint for developers to build NW

OSI ~: theoretical | **TCP/IP ~:** implemented in internet

PROTOCOL: Rules for data transmission agreed by sender, receiver. | E.g: data formats, error detection.

LAYERING: Simplifies model, standardised interface

TCP/IP [PROTOCOL] {PRO. DATA UNIT} <ADDRESSING>

Each layer only communicates with the adjacent layers.

At each layer, data unit formats; Headers with info in sender & removed from data unit in receiver's process

5 Application – High level FN to <u>end users</u> [HTTP/SMTP] {messages} <N/A>

Other protocols: FTP, SSH (file transfer) ; SMTP, POP3, IMAP4 (email) ; HTTP/HTTPS (webpage)

4 Transport – FN to transmit messages <u>btw 2 programs</u> [TCP/UDP] {segment/ datagram} <port number>

<u>*Transmission Control Protocol (TCP):*</u> uses 3 way handshake to est connection before transmission. Data is broken up into segments with sequential no. to reassemble at receiver. Ensures <u>security</u> & <u>validity</u> of data.

3 way handshake: [1] SYN [2] SYN/ACK [3] ACK

Sequencing: Data transmitted in segments, TCP places a seq. no. on EACH segment sent, so receiver can reorder.

TCP > UDP. Connection est. first -> used when high reliability necessary & transmission time is less critical.

<u>*Universal Datagram Protocol (UDP):*</u> Connectionless

+: No waiting time | **-:** Stability & reliability may be missing

UDP used when performance **>** receiving all data

<u>*Port no.:*</u> No. representing a process running on NW

<u>Socket:</u> Combination of IP address + port no. Identifier for an application process on NW

3 Network / Internet – FN to determine <u>route</u> btw 2 devices [IP] {packets} <ip address>

Internet Protocol (IP): responsible for routing individual datagrams and addressing.

IP Address: <u>logical address</u> that identifies devices on NW.

Network part: identifies LAN in which device is located

Host part: identifies the device in LAN

Device: Router – Uses IP addresses to send data packets to designated host. Maintains a routing table.

2 Data Link – FN to transmit packets between 2 devices in same NW. [Ethernet/WiFi] {Frames} <MAC address>

MAC: Media Access Control (physical address of device; unique for each machine)

Describes how devices format data for transmission to other devices

Device: Modem - Converts data from analog to digital (btw two transmission media); Switch – sends data frames to designated host, maintains a MAC table.

1 Physical – FN to transmit individual bits through a transmission medium [10 Base T/ 802.11] {Bits} <NA>

CLIENT-SERVER NW: one or more devices act as a server, other devices request service from server.

~ SERVER: Centralized storage area for resources; Provides service to other devices in NW, and is only dedicated to such tasks; Controls access to resources on NW, Filters network traffic

~ CLIENT: Sends request to server and server responds accordingly; Does not share any resources.

~ +: Faster traffic; more reliable – if 1 client down others still can use; Access rights of client is controlled; Resources secured on server and can be updated/ backup easily

~ -: If server down, whole network is down. Centralized server more exp to build, requires maintenance

---

PEER-TO-PEER NW: All devices interact and share resources with any other device without going through a server. They share the responsibility and capability.

MAIL SERVERS: Handle the transfer of messages to and from other mail servers and email clients.

SMTP: Send ; **POP3:** Receive & Store @ Client ; **IMAP:** Receive & Store @ Server, SYN on devices, Better org.

SMTP: SMTP server at sender's email service recognises the recipient's address' domain server, and connects to it with DNS. Message will then be handed to POP3

POP3: The protocol copies the emails and stores it locally

IMAP: Copies emails onto server so that multiple clients can access them. Client retrieves emails over TCP/IP connection

On the application layer of TCP/IP model.

DHCP SERVER: Dynamic Host Configuration Protocol

DHCP allocates unique IP address to clients in a NW.

Static Alloc^{n}: DHCP Server **passes** address to client

Auto Alloc^{n}: DHCP Server **assigns permanent** IP address to client from its pool of IP addresses.

Dynamic Alloc^{n}: DHCP Server **leases** a **reusable** IP to client for a period of time before it expires. (most often used)

On the network layer of TCP/IP model.

DHCP 4 STEP LEASING PROCESS

Discover: Client **broadcasts** to look for DHCP Server

Offer: DHCP Server offers an IP address

Request: Client requests for server to release address

ACK: DHCP server sends the address to client for ACK

DNS SERVER: Domain Name System

DNS translate domain names to IP addresses. E.g google.com -> 172.217.7.3

Hosts can query the database to be pointed towards the necessary IP address. It is implemented in a hierarchy.

E.g -> ROOT --> Top Level Domain (.com) ---> authoritative (google.com) ----> subdomain (www.google.com)

ENCRYPTION: Process of **encoding** a message in such a way that **only authorized parties can access it**

Sender: uses a secret key and encryption algo to encrypt plaintext into a cipher.

Receiver: uses a secret key and decrypt algo to descript cipher back to og plaintext.

SYMMETRIC ENCRYPTION: Same single encryption and decryption key that is shared between sender and receiver.

+: Fast processing

-: Low security if the key is intercepted

Standard: AES (Advanced Encryption Standard)

ASSYMETRIC ENCRYPTION: One **public** key and one **private** key that are mathematically related. One cannot be derived from another. | RSA (Rivest-Sharmir-Adleman)

Sender uses receiver's public key to encrypt message. The receiver's private key is the only key that decrypts msg.

+: Even with public key, message not leaked ; More secure

-: Slower than symmetric encryption

DIGITAL SIGNATURE (DS): Encryption and Decryption technology that **secures data associated** a signed document and **verifies the authenticity** of the document.

Authentication: Ensures message from known sender

Non-repudiation: Sender cannot deny that they sent it

Integrity: Message is not altered in transit

Usage: Used on emails, internet, bloc chains.

Technologies: Hashing ; Encryption (Priv + Public Keys)

Hashing: Generates a short unique text string, acts as a unique digital fingerprint. | SHA256 (Secure Hash Algo)

~ (SENDER): The sender uses a hash algo to create hash. Private key encrypts hash to DS. Msg & DS sent to receiver.

~ (RECEIVER): The receiver uses the sender's public key to decrypt the DS back to the sender's hash.

IDENTIFICATION: user claims an identity.

AUTHENTICATION: Users proving their identity by providing credentials. System validates the identity of the user. E.g: Password, Fingerprint, Retina, Secret Qns, OTP

MULTIFACTOR ~: Sender public key to decrypt the digital signature back to the sender's version of hash. Receiver uses the same hash algo to create new hash from msg. If hashes match, the data is not altered.

AUTHORISATION: System validates the granted the permission of an authenticated user to access resources.

DATA VALIDATION: Chk data format valid; Length checks

DATA VERIFICATION: Check that data is what user intends to give. E.g double input of password.

MALWARE: Malicious software that is harmful to the computer system. Can alter/disable/destroy/steal data from computer, mobile devices and the network.

Virus: Attaches itself to a file/program ; Remains dormant until executed ; Replicate to infect other computers

Worm: No need to be attached to files ; Self-replicating.

Trojan Horse: Appears as legit program ; Once it gains the access into the computer, it runs malicious code that damages the computer. Does NOT replicate itself.

Ransomware: Locks computer, encrypts the data ; Forces the user to pay a ransom to get data back.

DENIAL OF SERVICE (DOS): Attacks network traffic to exhaust resources, cannot fulfil legitimate requests.

DDOS (DISTRIBUTED DOS): Multiple compromised devices (using botnets) to attack systems.

SOCIAL ENGINEERING ATTACK: Use of deception and trickery to convince users to compromise their systems. *Phishing:* attacker sends email address that impersonates a company, asking for bank account details

Spoofing: attacker conceals identity, IP/MAC address

Spam: users flooded w messages full of ads and viruses.

PROTECTION SCHEME: Update OS, install antivirus, do not click suspicious links, do not connect to public WiFi

Firewall: **monitor and control** all in/outcoming NW traffic, but hackers can bypass and does not block internal attacks

Proxy Server: Acts as intermediary for req from clients, hides IP address and details, control in/out NW traffic

VPN: Adds encryption on any data transmitted in and out

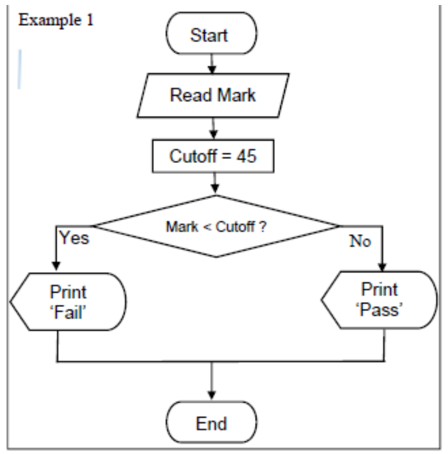
IDS: Scans, monitor NW traffic, sounds alert but no action.

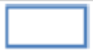
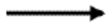
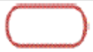
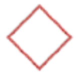
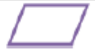
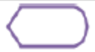
Signature Based: Pre-defined set of rules to identify traffic. **Limitation**: cannot detect unknown attacks.

Anomaly Based: Database of unacceptable traffic patterns. **Limitation:** inaccurate profile, difficult to determine cause.

IPS: IDS, but **blocks** unauthorised access. May drop packets, reset connections, sound alerts.

FLOWCHARTS



Symbol	Symbol Name (alias)	Symbol Description
Process / Operation Symbols		
	Process	Show a Process or action step. This is the most common symbol in both process flowcharts and business process maps.
Branching and Control of Flow Symbols		
	Flow Line (Arrow, Connector)	Flow line connectors show the direction that the process flows.
	Terminator (Terminal Point, Oval)	Terminators show the start and stop points in a process. When used as a Start symbol, terminators depict a *trigger action* that sets the process flow into motion.
	Decision	Indicates a question or branch in the process flow. Typically, a Decision flowchart shape is used when there are 2 options (Yes/No etc.)
Input and Output Symbols		
	Data (I/O)	The Data flowchart shape indicates inputs to and outputs from a process. As such, the shape is more often referred to as an I/O shape than a Data shape.
	Display	Indicates a process step where information is displayed to a person (e.g., PC user, machine operator).

COMPUTATIONAL THINKING

Decomposition: breaking down a complex problem or system into smaller, more manageable parts

Pattern Recog.: look for similarities within problems

Abstraction: focus on impt info, ignore irrelevant detail

Algorithms: developing a step-by-step solution to the problem, or the rules to follow to solve the problem

PSEUDOCODE

DECLARE: Creation of variable in memory
CONSTANT: Special variable that cannot be changed.
ASSIGNMENT: <identifier> **<--** <value>
INPUT <identifier>: (collect user input)
OUTPUT <values>: displays values

```
IF <condition>
  THEN
     <statement>
  ELSE
     <statement>
ENDIF
```

```
FOR INDEX = 1 TO 5:
      Extract <- Extract + Sentence[Index]
ENDFOR
```

```
WHILE <condition> DO
    <statement>
ENDWHILE
```

```
PROCEDURE <identifier>
   <statement>
ENDPROCEDURE
```

```
FUNCTION <identifier> RETURNS <data type>
   <statement>
   RETURN <variable>
ENDFUNCTION
```

FILES

```
with open('myfile.txt','r') as f:
    data = f.read()
```

STRINGS

```
startswith(substring)       Returns True/False
endswith(substring)         Returns True/False
```

`find(substring [, start , end]):`
Returns the lowest index in string of substring, else -1.
`replace(old, new, count):`
If count given, only first count appearance replaced.

`count(substring [, start , end]):` returns the number of occurrences of substring in the string.
`split(sep):` Return the list of the words in the string, using sep as the splitter. If empty, uses whitespace.

DATABASE (DB): Organized collection of structured info

DATA REDUNDANCY: Same data stored more than once

DATA INTEGRITY: Data should not be entered twice (data validity), Data needs to be changed across all fields.

FILE - BASED (FB) APPROACH: Sharing file (server/ email..)
-: Lack of privacy; data not suitable; hard to update; data dependency issues (compatibility btw software)

DB APPROACH: External, Conceptual, Internal Levels
+: Multiple (user) views; Won't affect other user; Insulation between program & data; multiuse transaction processing, program-data independence.
DB TYPES: Relational (tables, columns, rows); NoSQL (non-relational); Flat-file; hierarchical; network.

DB vs FB: Atomicity – changes are performed as single operation | **C**onsistency – data in consistent state | **I**solation – Intermediate state of transaction is invisible | **D**urability – Changes to data persists, even if sys failure.

DBMS: Interface between database and end users or programs, allowing users to retrieve, update, manage how the info is organized, and optimized.
-: High initial investment (additional hardware, manpower); When not necc; When Data is too complex

RELATIONAL DATA MODEL: Organizes & represents data in form of a table. (Based on math. concept of relation)

RELATION (TABLE): Table with rows and columns

TUPLE (ROW): Row of a relation

ATTRIBUTE (COLUMN): Named column of a relation

DOMAIN: Set of allowable values for attribute

CANDIDATE KEY: Minimal attributes that identify tuples

PRIMARY KEY (PK): Candidate key to identify tuples UNIQUELY. | Criteria: less likely to have its values changed/ lose uniqueness; with fewest characters; min. set of attributes, easier to use from user's perspective.

SECONDARY KEY: Candidate key that are not PK.

FOREIGN KEY: Attribute that provides logic link btw tables.

COMPOSITE KEY: Combination of two or more attributes.

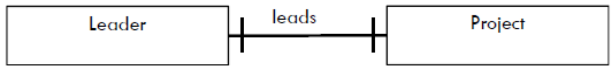
ENTITY RELATIONSHIP DATA (ERD) MODEL: Used for data modelling for databases (easy to discuss and explain)

ENTITIY: Tables that hold specific info (rep by rectangles)

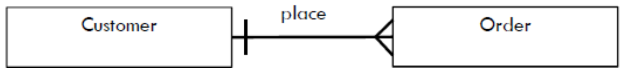
RELATIONSHIPS: Assoc. / interac^{n} btw entities (rep by line)

CONNECTIVITY: One to One; One to Many; Many to Many

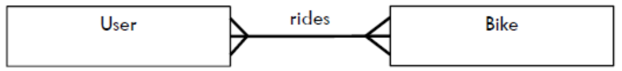
1-1: Each project 1 leader | Each leader leads 1 project



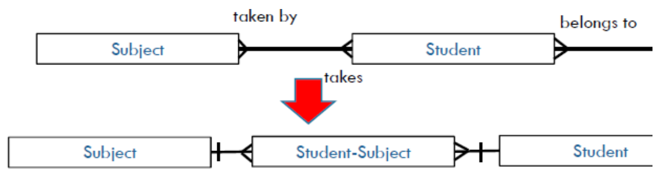
1-Many: User place many orders | Each order 1 user.



Many-Many: User got many bikes | each bike many users.



REFINEMENT: (Split the M:M R/S into 2x of 1:M R/S)



DATA REDUNDANCY (DR): When data is being stored more than once. Causes data anomalies.

DATA ANOMALIES: When not all changes successful

1. Insertion anomaly: Until class created, student data cannot be inserted, unless it is set as NULL
2. Update anomaly: If a class name is changed, all records need to be updated, else there is data inconsistency
3. Deletion anomaly: If student table has student and class info, when table deleted, class info lost too

FUNCTIONAL DEPENDENCIES (FD): Eliminate redundancy & anomalies; ensures all attributes in a table belong there. If you derive Y from X, Y is functionally dependent on X.

NORMALIZATION: New DB: Analyse R/S among attributes in ERD to improve. | Existing DB: Analyse R/S among attributes in data structure to improve DB structure

1st NORMAL FORM (NF): Each attribute has ONE value.

STUDENTID	NAME	SUBJECT	
~~0001~~	~~Daren~~	~~MA, CP~~	NOT 1NF
0001	Daren	MA	1NF
0001	Daren	CP	

2ND NF: 1NF + All non-key attributes FD on (composite) PK.

SCOREID	STUDENTID	MARKS	SUBJ_ID	TEACHER
1	0001	70	1	Ms MA
2	0001	30	2	Mr CP
3	0002	70	1	Ms MA

** In this case, STUDENTID & SUBJ_ID are composite PK. TEACHER is only dependent on SUBJ_ID. Hence, it is not FD on composite PK. To remove partial dependency, create new SUBJ_ID table (SUBJ_ID, TEACHER NAME)

STUDENT TABLE

SCOREID	STUDENTID	MARKS	SUBJ_ID (FK)
1	0001	70	1
2	0001	30	2
3	0002	70	1

SUBJECT TABLE

SUBJ_ID	TEACHER_NAME
1	Ms MA
2	Mr CP

3RD NF: 2NF + Not Transitive Dependent (TD).

SCOREID	STU_ID	SUBJ_ID	MARKS	EXAM_NAME	TOTALMARK
1	0001	1	70	Lab	100
2	0002	2	30	Theory	50

TD: SCOREID -> EXAM_NAME -> TOTALMARK
SCOREID -> TOTALMARK is a TD
Put EXAM_NAME, and TOTALMARK into another table with PK EXAM_ID, use EXAM_ID as Foreign Key in SCORE.

STUDENT TABLE

SCOREID	STU_ID	SUBJ_ID	MARKS	EXAM_ID(FK)
1	0001	1	70	9931
2	0002	2	30	9922

EXAM TABLE

EXAM_ID(PK)	EXAM_NAME	TOTAL_MARK
9931	Lab	100
9922	Theory	50

SQL: Create database + structures | Basic data mgmt. | Complex queries to change data into useful info

DATA DEFINITION LANGUAGE: Define + Create DB (objs)

1. CREATE: Create DB| For CREATE TABLE, check below
   `CREATE DATABASE` *`database-name;`*
2. ALTER: Alter DB structure
   ```
   ALTER TABLE Classes
   RENAME TO Class
   ADD ClassID INTEGER (note: run once at a time)
   ```
3. TRUNCATE: Remove ALL records
   `DELETE FROM` *`table-name;`*
4. DROP: Delete DB, Table, Objects
   `DROP TABLE [IF EXIST]` *`table-name;`*

```
FOREIGN KEY CONSTRAINT:
CREATE TABLE STUDENT (
StudentID INTEGER PRIMARY KEY AUTOINCREMENT,
ClassID INTEGER,
Name TEXT(100) NOT NULL,
NRIC TEXT(15) NOT NULL,
FOREIGN KEY(ClassID) REFERENCES
Class(ClassID) );
```

DATA MANIPULATION LANGAUGE:

1. SELECT

   SELECT *expressions* FROM *table* WHERE *conditions*
   Use * as expression to select ALL | Conditions can be like ‘%TAN%’ | %: matching >= 0 characters | “BETWEEN 5 and 20” | BETWEEN is inclusive of 5 & 20

   ```
   SELECT Todo.id, Category.name, Todo.description
   FROM Todo INNER JOIN Category ON
   Todo.categoryName = Category.id
   ```
2. INSERT

   INSERT INTO *table*(columns) VALUES(values list)

   ```
   INSERT INTO Category(Name,Description)
   VALUES(‘Stationary’,’Writing Materials’)
   ```
3. UPDATE

   UPDATE *table* SET *column_name = column_value ..*
   WHERE *conditions*

   ```
   UPDATE Category SET Name=’Stationary’ WHERE id>5
   ```
4. DELETE

   DELETE FROM *table* WHERE *conditions*

   ```
   DELETE FROM Category WHERE id=5
   ```

Examples: ‘SQL ALL COMMANDS.sql’, ‘SQL Exercise 1’

SQLITE3 (PYTHON)

PYTHON CODE SNIPPET

```
import sqlite3
db = sqlite3.connect(‘database.db’)
db.row_factory = sqlite3.Row
cursor = db.cursor()
cursor.execute(“SELECT * FROM Category”)
category = cursor.fetchall()
```

NOTE: Without line 3, what returned is *a list of tuples of the column data* within. With line3, what returned is a *list of dictionaries with key column_name and value column_value*

PYTHON INSERT INTO

```
db.execute("""INSERT INTO
Todo(categoryName, description)
VALUES (?,?);""", (category,description))
db.commit() #this is impt
```

NOTE: Using triple “”s, we can escape all brackets within query. | The ‘?’s mark variables within python that we want to insert into the query. | Notice the brackets, and that python variables are stored in a tuple.

MONGODB (NOSQL):

WHY: ~ is a non-relational database (no predefined schema). | Document database | Horizontally scalable.
+: Dynamic schema, Fast data storage, Handle large data
WHY SQL: Data has fixed schema | Complex & Varied queries frequently performed | Atomicity, consistency, isolation, durability (ACID) is impt | Supports high number of simultaneous clients

```
from pymongo import MongoClient
client = MongoClient(‘localhost’,27017)
mydb = client[‘database’]
mycoll = client[‘customers’]
bookList = [{‘bookid’:234},{‘bookid’:331}]
x = mycoll.insert_many(bookList)
print(x.inserted_ids)

query = {‘bookid’:{“$gte”:100, “$lte”:200}}
q2 = {‘desc’:{‘$regex’:‘hi’}}
q3 = {‘desc’:’hi’}
q4 = {'$or':[{'borough':"Staten"},
{'borough':"Queens"}]}
q5 = {'borough':{'$in':["Staten", "Queens"]}}
q6 = {"address.coord":{$type: 1}}

projection = {“_id”:0}
result = mycoll.find(query, projection).sort(‘bookid’,1)

mydb.collection.update_many({“bookid”:{“$exists”:False}},
{“$set”:{“bookid”:000}})
```

regex query has to be in dict | address.coord – dict of values within another dict | .sort(‘borough’,1) or -1
diff btw q2 & q3: q3 is exact, q2 ignores trailing char
common regex queries: `$eq,$ne,$gte,$gt,$lte,$lt, $in(q5),$nin| $and,$not,$or(q4)| $exists| $set, $unset| $type:1/2/3/4(double/str/obj/array) (q6)`

```
# THIS IS CLIENT CODE
import socket
s = socket.socket()
s.connect(('localhost',9999))
data = b''
send = ''
while send != 'end':
    send = input('Input CLIENT -> ')
    s.sendall(send.encode())
    data = s.recv(1024)
    print(f"Received from SERVER: {data.decode()}")
s.close()
```

```
# THIS IS SERVER CODE
import socket
s = socket.socket()
s.bind(('localhost',9999))
s.listen()
s, address = s.accept()
data = b''
send = ''

while True:
    data = s.recv(1024)
    print(f"From connected user: {data.decode()}")
    if data == b'end':
        break
    send = input("Input SERVER -> ")
    s.sendall(send.encode())

send = "'Ok, then. bye!'"
print("INPUT SERVER ->", send)

s.sendall(send.encode())
s.close()
```

FLASK:

```
import flask, sqlite3, os
from flask import *
from werkzeug.utils import secure_filename
app = flask.Flask(__name__,
template_folder="TASK4_3")

@app.route(‘/’, methods=[‘GET’,’POST’])
def home():
    if request.method == 'GET':
        return render_template('home.html')
    else:
        name = request.form['name']
        temp = request.form['temp']
        date = request.form['date']
        try:
            photo = request.files['photo']
             filename=secure_filename(photo.filename)
             imageURL= os.path.join('uploads', filename)
             photo.save(imageURL)
        except:
            filename = None
     render_template(‘student.html’, name=name)

@app.route('/uploads/<filename>')
def getFile(filename):
    return send_from_directory('uploads', filename)
app.run()
```

DATA PRIVACY: The requirement for data to be accessed by or disclosed to authorised persons only.
CONSUMERS: More control, easier access and correction of data, reduced unsolicited telemarketing messages.
BUSINESSES: Increased consumer confidence, Easier cross-border transfer, enhanced efficiency, branding & competitiveness.

PDPA (PERSONAL DATA PRIVACY ACT): Governs the **[1]** collection , **[2]** use and **[3]** disclosure of **[4]** personal data [PD] by organizations in a manner that recognizes both the right of individuals and the need of organizations to collect, use or disclose **[5]** PD for purposes that a reasonable persons would consider appropriate in the circumstances.

PURPOSES OF PDPA
PDPA takes into account – consent, notification, appropriateness, accountability.
1. Collection: actions through which org. obtains PD
2. Use: actions through which org. uses PD
3. Disclosure: actions through which org discloses/ transfers/ makes available any PD.
4. PD: data of an individual who can be identified from that data.
5. Purpose: objective/ reasons relating to PD use.

DATA OBLIGATIONS
1. CONSENT OBLIGATION – Only disclose PD for which an indiv. who has given his/ her consent
2. PURPOSE LIMITATION OBLIGATION – may collect data about indiv. for purposes a reasonable person would consider appropriate + indiv. Has given consent
3. NOTIFICATION OBLIGATION – notify indiv. of purposes for which the org is intending to collect/ use data on / before collection
4. ACCESS & CORRECTION OBLIGATION – PD of an indiv. and info about how it has been used/ disclosed within a year before the request should be provided. However, orgs are not allowed to give access if info could reasonably be expected to cause harmful effect. Orgs are also required to correct any error in PD
5. ACCURACY OBLIGATION – ensure data is accurate
6. PROTECTION OBLIGATION – make reasonable security arrangements to protect the PD that your orgs possesses or controls to prevent unauthorised access, collection, use, disclosure…
7. RETENTION LIMITATION OBLIGATION – cease retention of PD or remove the means why which data can be associated if data is not necessary for any business or legal purposes
8. TRANSFER LIMITATION OBLIGATION – Transfer PDPA to another country only according to requirements stated by regulations, ensure that the standard of protection of data is the same as PDPA.
9. ACCOUNTABILITY OBLIGATION – Make information about data protection policies, practices available on request. Designate a Data Protection Officer to ensure organization complies with PDPA guidelines.

DO NOT CALL REGISTRY (DNC): One can register in the registry to prevent oneself from getting unnecessary marketing calls as PDPDA prohibits organisations from sending marketing messages to SG telephone numbers that are registered with the DNC.

3 DNC REGISTERS: No Voice Call/ No Text Message/ No Fax Message.

NRIC (SG NATIONAL REGISTRATION IDENTIFICATION CARD): It is a unique identifier assigned to SG citizens and permanent residents. They are permanent, irreplaceable and used in a variety of government transactions.

COLLECTION OF NRIC: Organizations (excluding governments) are only allowed when NRIC is required under the law or when NRIC is necessary to accurately establish / verify the identities of the individuals (e.g in a clinic).

Instead of NRIC, Participants can be identified with their mobile number, or be asked to give the last 4 characters of the NRIC (i.e. partial NRIC) for verification purposes.