ASCII - each character (1 byte)

GB for simplified (2 byte)

Big5 for traditional (2 byte)

Unicode - any language (1-4 byte)

**Data input error:**

* Data source error - Input incorrect
* Transcription error - oO il
* Transposition error - 12 21

Field presence check </>

Field length check <--->

Range check <1...2>

Fixed value check <M/F>

Format check <@gmail.com>

type check <str>

Check digit <parity check>

| Overflow: 01 + 01 = 10 (none) 11 + 01 = 100 (overflow)

| Odd: 11100000 3 x 1: odd matches odd ∴ parity bit = 0

| Even: 11100000 3 x 1: odd not even ∴ party bit = 1

| // add a “1” to match the even requirement

| Sent odd: 11100000 3 x 1: odd matches odd = error-free

| Sent even: 11100000 3 x 1: odd not even = error

**Two’s compliment:** Invert & add one. (current 4 bit memory unit)

* -2 🡪 0010 (2) 🡪 1101 🡪 1110 (-2)

**Data Hierarchy:** field - record - table - database

**Color depths:**

* 2^1 colors (bit:1) - black & white
* 2^8 colors (bit:8) - monochromatic
* 2^24 colors (bit:24) - RGB (true color)

[/F] Size/ppi | size/dpi

[/F] vector graphics - independent of resolution & color depth, simple line arts & charts

Analogue data - can’t be stored

Digital data - can be stored

[/F] Quantization - sample data > digital code with predefined scale

[/F] Sampling - Store analogue data sample

**Office automation:**

^ Productivity

+ data management

+ communication

V company expense by saving storage space

|  |  |
| --- | --- |
| **Expression** | Info |
| =A1 | Get cell val A1 |
| =$A$1 | Get the absolute cell val A1, copying doesn't affect it |
| =Name!... | Get cell val from Sheet with <Name> |
| ="Yes"A1 | Get cell val A1 with string "Yes" in front |

[/F] Sorting is a process of reordering the records according to a specific criterion

[/F] Filtering is a function to select & display the records meeting the criteria specified by a user, hiding the other records

[/F] Scenario Manager allow users to create different scenarios & compare the results by switching between them

[/F] Goal seek works reversely to evaluate the values of cells concerned from a given conclusion

**Pivot table guide: (Ref P.1-280)**

|  |  |
| --- | --- |
| **Filters** | **Columns** |
| Use filters to filter out certain value of certain groups | Stuff to display on top of table |
| **Rows** | **Values** |
| Stuff to display on the left  When things are stacked up, the topper item becomes the group of the bottom items | Values to display in the table  Types can be:   * Sum * Count * Average / Max / Min |
| **Example:**  | House |  | Student ID | | **Output:**  [-] <House>  10223  [-] <House2>  32321  39211 |

**DBMS - Microsoft Access**

|  |  |
| --- | --- |
| **Field** | Specified by field name, data type & field length |
| **Key field** | Unique to a specific record |
| **Input mask** | Prevent user from inputting invalid data |
| **Datasheet view** | Display a table organized in rows & columns |
| **Design view** | Display underlying structure of a table |
| **Forms** | Graphical UI used to edit & display data |
| **Reports** | Print data in a professional & easy-to-read layout |

**Query Commands:**

|  |  |
| --- | --- |
| **SELECT** | fields(s) - target fields to be used as results |
| **FROM** | table(s) |
| **WHERE** | condition - the condition of a field in the same row  WHERE column=’value’; WHERE column>1; |
| **ORDER BY** | field(s) |
| **ASC / DESC** | Order by ascending or descending |
| **;** | Marks the end of the command |

**CPU**

* Components: ALU Arithmetic & logic unit, CU control unit, registers
* **ALU**

Performs logical operations

* **CU**

Keeps track of the sequence of instructions being processed

* **Registers**

Registers are memory units inside CPU providing storage space for ALU CU

—————————————————————————————

GPRs generate purpose registers:

1. Accumulator AX
2. Base register BX
3. Counter CX
4. Data register DX

Functions

* + - Add AX, AX: Double the value of AX
    - Add AX, 1: Add 1 to AX (LOAD)
    - Sub AX, AX: Set AX to 0
    - Sub AX, 1: Minus 1 from AX (CMP)
    - Div AX, AX: Set AX to 1
    - Div AX, 1: Divide AX by 1

—————————————————————————————

Status register

* + - Zero sign: “0”:”1”?”0”
    - Sign flag: “<0”:”1”?”0”

—————————————————————————————

Control registers

* Instruction register | Instructions to execute by CPU
* Program counter | Memory address of next instruction
* MAR memory address register | Address of memory location to / from data to be transferred
* MDR memory data register | Data to be written into / read from the location

—————————————————————————————

* **Sys bus**

Physical wiring that connects the components of computer Sys

64-bit data bus 🡪 transmit 64 bits data at a time

Computer processor bit = data bus bit

—————————————————————————————

1. Data bus | Transfer **data** & **instructions**
2. Address bus | Transfer **source** **address** or **destination** **address** of data
3. Control bus | Indicate direction of data transfer & coordinate the timing

| of data transfer

**Machine cycle**

Process of executing an instruction in CPU

1. Fetch | Read next instruction from memory to IR
2. Decode | Identify operation code & oper&s
3. Execute | Perform required operation
4. Interrupts | Check for & h&les interrupts, CPU saves process status & h&les

| interrupt, then resume operation of next instruction

**Measure CPU speed**

* **CPU speed calculation**

1 Hz = 1 clock cycle / sec

Time taken to complete operation =

Time taken for running program =

* **Word length**

Number of bits of data & instructions the CPU can handle at one time

referenceMaterials

b bit Hz Hertz

B 8 bits kHz 1,000 Hertz

KB 1024 Bytes 210 MHz 1,000,000 Hertz

MB 10242 Bytes 220 GHz 1,000,000,000 Hertz

GB 10243 Bytes 230

TB 10244 Bytes 240

Colors Transparent Alpha Compression Scaling Animation

PNG 224 1 1 0 0 0

JPG 224 0 0 1 0 0

GIF 28 1 0 0 0 1

SVG 224 1 1 0 1 1

BMP 28 1 1 0 0 0

Styles Size Open Graphics Edit

PDF 1 Big Special 1 Hard

TXT 0 Smol Any 0 Easy

RTF 1 Smol Any 0 Easy