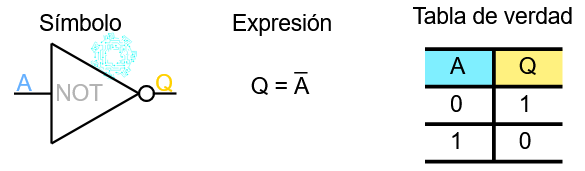
**DICCIONARIO**

# Nivel 1 (Turing)

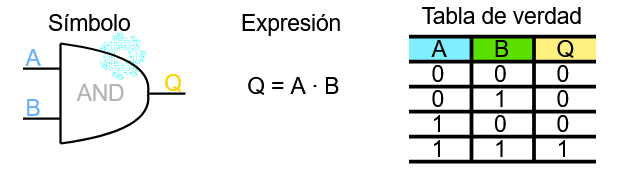
## TAREA 12

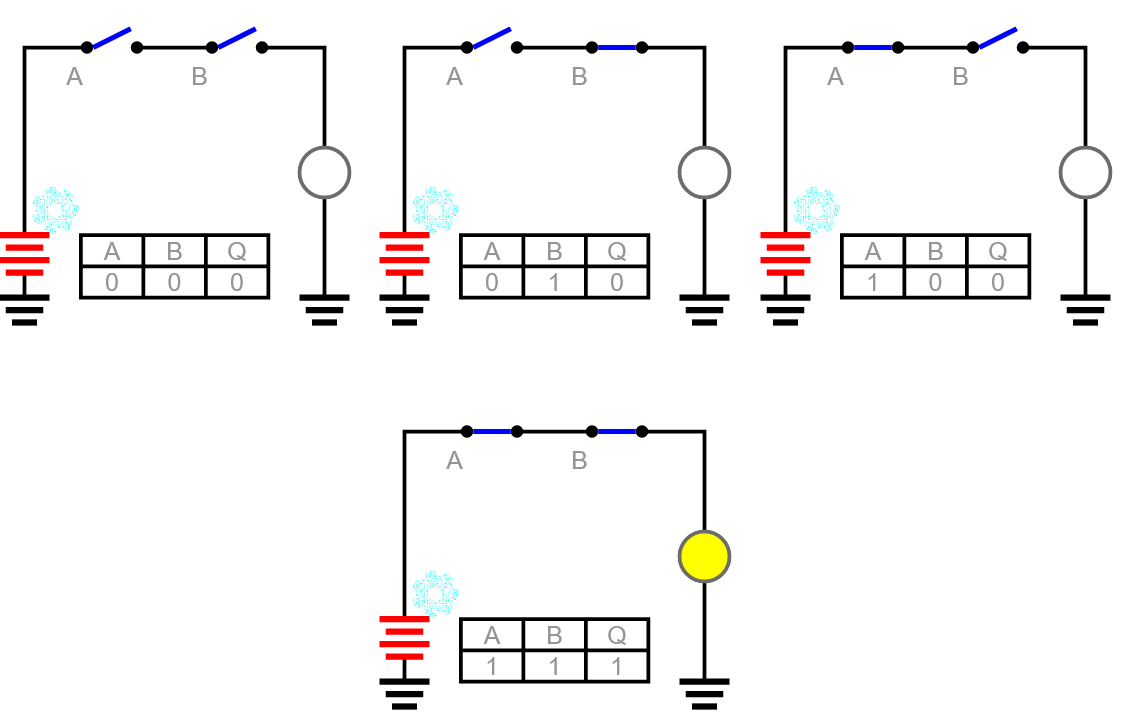
TRANSISTORES

* Tipos
  + Compuerta NOT/INV
    - Su propósito es producir una salida inversa o contraria a su entrada
    - 1: activo
    - 0: inactivo

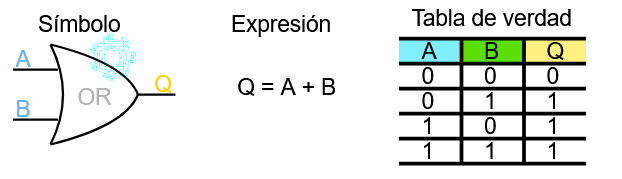


* + Compuerta AND
    - Todo o nada
    - La expresión del medio de la foto es Álgebra de Boole
    - Para tener una salida activa es necesario que sus entradas tengan un estado binario 1

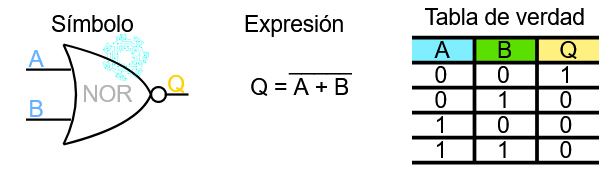




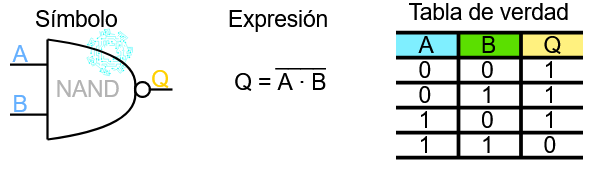
* + Compuerta OR
    - Cualquiera o todo
    - Esta compuerta se encuentra en estado activo siempre y cuando una de sus entradas tenga un estado binario activo “1”



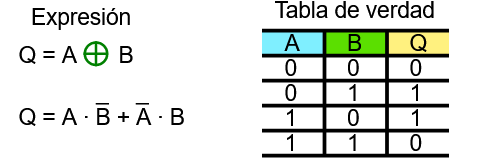
* + Compuerta NOR
    - Versión inversa de la compuesta OR

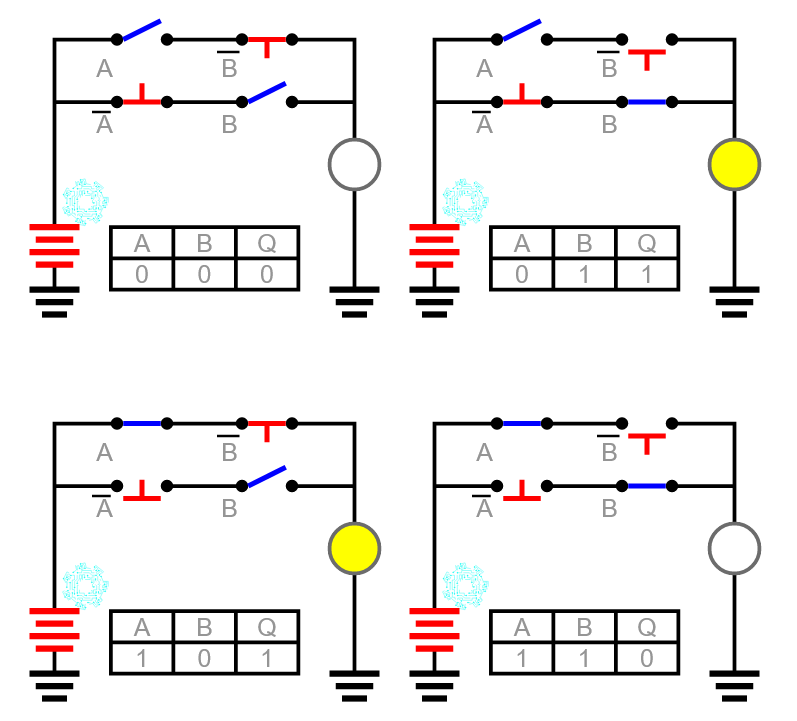


* + Compuerta NAND
    - Versión inversa de NAND

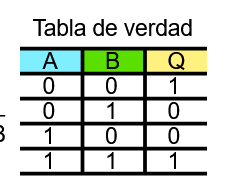


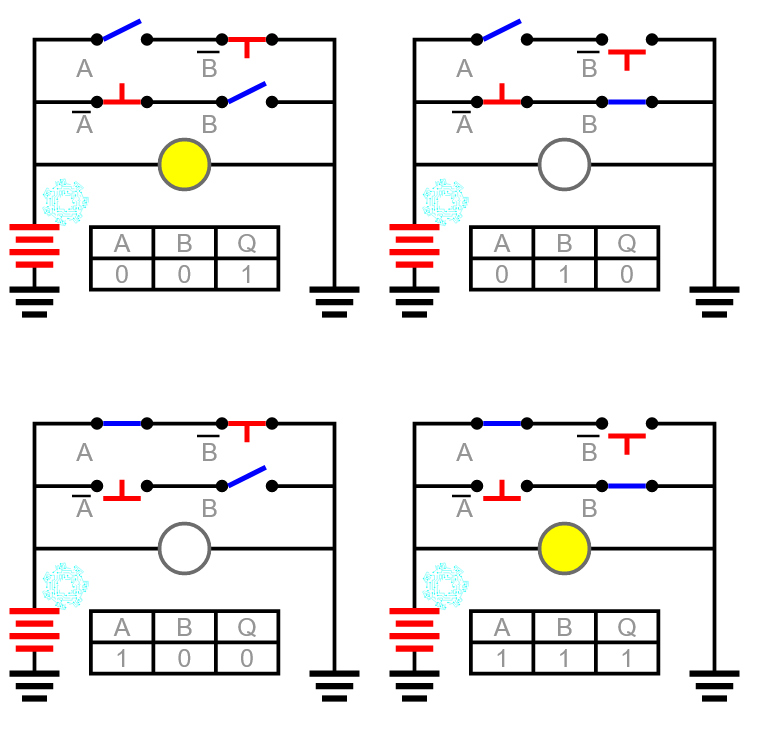
* + Compuesta XOR
    - Algunos pero no todos
    - La salida tiene un estado activo “1” al tener las entradas en estados diferentes





* + XNOR
    - Negación de la compuerta XOR
    - Cuando las entradas sean iguales se representará una salida en estado “1” y si son diferentes la salida será un estado “0”

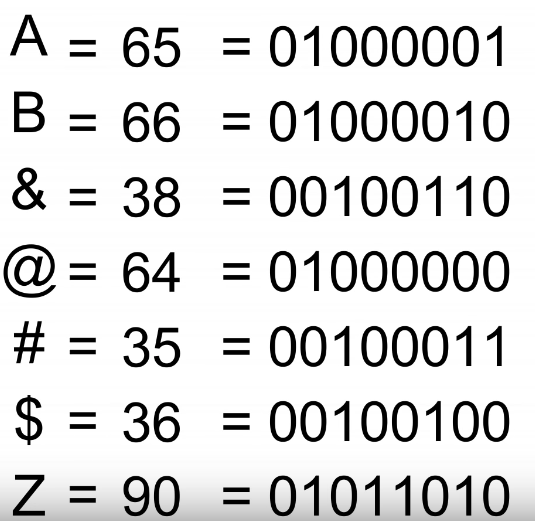


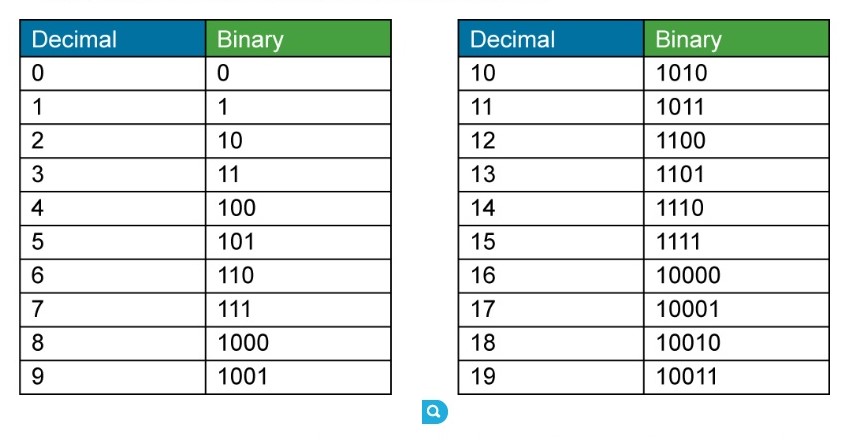


* CALCULADORA
  + Números binarios
    - Las posiciones son potencias de 2.

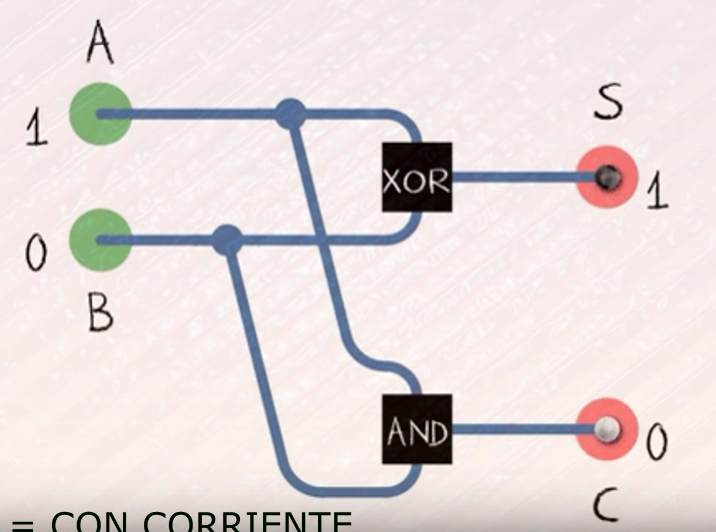


* + - Las letras y símbolos tienen asociados unos números para que un ordenador pueda entender nuestro idioma

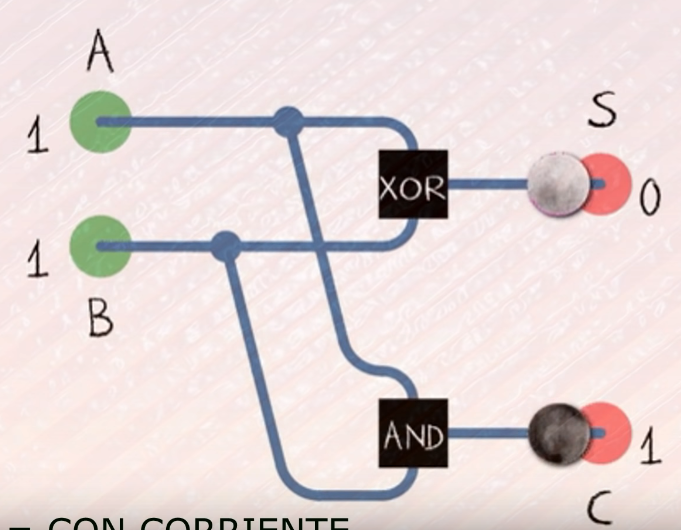




* + 1+0



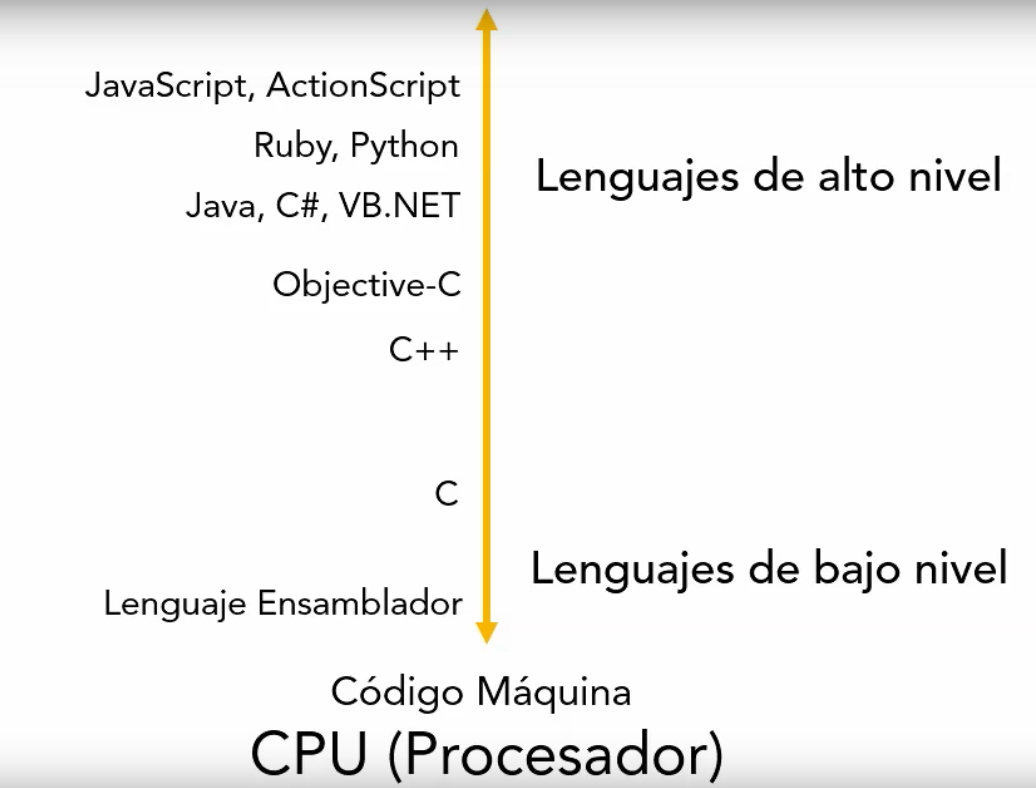
* + 1+1



* Bit y Byte
  + Bit
    - Binary digit (digito binario)
    - Un bit tiene solamente un valor= 0 o 1
    - Varios bits combinados entre sí dan origen a otras unidades: “byte”, “mega”, “giga”, “tera”
  + Byte
    - Es un conjunto de 8 bits
    - Se puede hacer una correspondencia biunívoca entre cada número decimal (0 a 9), las letras mayúsculas y minúsculas (A hasta Z), los símbolos matemáticos, la puntuación, y demás símbolos, con un respectivo byte.
* CPU 32 bits/64 bits
  + Los sistemas de 32 bits almacenan sus datos en piezas de 32 bits
  + Los sistemas de 64 bits almacenan sus datos en piezas de 64 bits
  + Al trabajar con piezas más grandes, puede hacer más en menos tiempo

## 1.2. Tarea 14

* Un lenguaje de programación lo componen una lista de instrucciones que organizadas de una manera conforman programas
  + Alto nivel
    - Se caracteriza por expresar los algoritmos de una manera adecuada a la capacidad cognitiva humana, en lugar de la capacidad con que los ejecutan las máquinas.
    - Estos lenguajes permiten una máxima flexibilidad al programador a la hora de abstraerse o de ser literal
    - Tendrán que ser compilados o interpretados para transformar el código fuente en código de maquina (código binario)
    - Ej.: Python, R, C++…
  + Bajo nivel
    - Es aquel en el que sus instrucciones ejercen un control directo sobre el hardware y están condicionados por la estructura física de las computadoras que lo soportan
    - Mínima abstracción entre el lenguaje y el hardware
    - Dos tipos:
      * Lenguaje ensamblador
        + Es el primer intento de sustitución del lenguaje de máquina por uno más cercano al utilizado por los humanos
        + Un programa escrito en éste lenguaje es almacenado como texto
        + Ej: MOV AL, 61h (asigna el valor hexadecimal 61 al registro “AL”)
      * Lenguaje máquina
        + Es el más primitivo de los lenguajes y es una colección de dígitos binarios o bits (0 y 1) que la computadora lee e interpreta.
        + Ejemplo: 10110000 01100001



## 1.3. Tarea 15

* Frontend
  + Frontend es la parte de un programa o dispositivo a la que un usuario puede acceder directamente.
  + Son todas las tecnologías de diseño y desarrollo web que corren en el navegador y que se encargan de la interactividad con los usuarios
  + HTML, CSS Y Javascript
* Backend
  + Backend es la capa de acceso a datos de un software o cualquier dispositivo, que no es directamente accesible por los usuarios.
  + Además contiene la lógica de la aplicación que maneja dichos datos.
  + También es el que accede al servidor, que es una aplicación especializada que entiende la forma de como el navegador solicita cosas.
  + Python, PHP, Ruby, C#, Java
* Librería
  + Una librería es un programa que resuelve algo, desarrollado generalmente por terceros, que se pueden reutilizar para no tener que hacer todo desde el principio.

## 1.4. Tarea 22

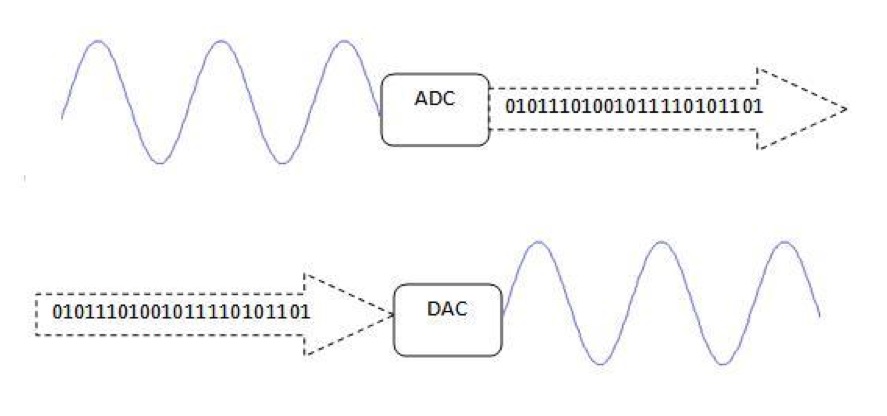
* Programación dinámica

## 1.5. Tarea 23

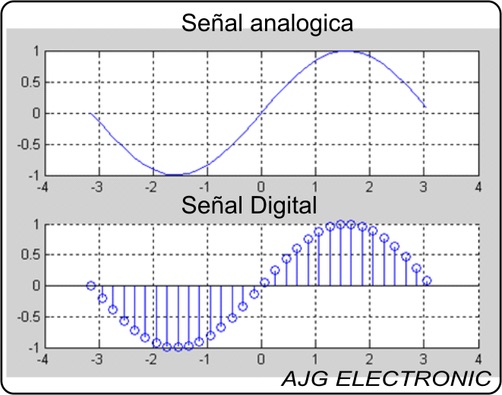
* Cifrado
* Descifrado

## 1.6. Tarea 24

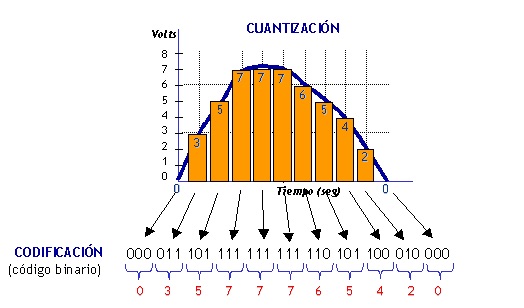
* Digital
  + Un sistema digital es cualquier sistema que permita crear, decodificar, transmitir o guardar información que se encuentra representada en cantidades tan restringidas que sus señales de entrada y salida solo admiten valores discretos.
    - Una variable discreta es una [variable](https://es.wikipedia.org/wiki/Variable_estad%C3%ADstica) que no puede tomar algunos valores dentro de un mínimo conjunto numerable, quiere decir, no acepta cualquier valor, únicamente aquellos que pertenecen al conjunto. Como ejemplo, el número de animales en una granja (0, 1, 2, 3, 4, 5, 6, 7, …). Otro ejemplo sería el número de hijos en una familia (1; 2; 3; 4; …).
    - Una variable continua toma valores a lo largo de un continuo, esto es, en todo un [intervalo](https://es.wikipedia.org/wiki/Intervalo_(matem%C3%A1tica)) de valores. Como ejemplo, la estatura de una persona (1,72 m; 1,719 m; 1,718 6 m....).
  + Ventajas de los sistemas digitales
    - Menor tamaño: los dispositivos basados en sistemas digitales tienden a hacerse cada vez más pequeños.
    - Eficiencia: la información digital se almacena rápidamente y en cantidades cada vez más grandes.
    - Precisión: como los sistemas digitales solo admiten valores discretos, son mucho más precisos.
    - Diseño: los dispositivos basados en sistemas digitales tienden a hacerse cada vez más fáciles de diseñar
    - Estabilidad: los sistemas digitales son menos susceptibles al ruido, es decir, a todas las posibles perturbaciones de la señal.
  + Desventajas de los sistemas digitales
    - Conversión: las naturaleza de las variables físicas es analógica (sonido, temperatura, distancia, peso) por lo tanto, es necesario usar un conversor para transformarlas en datos digitales.
    - Ancho de banda: la transmisión de señales en un sistema digital requiere de un ancho de banda mucho mayor que un sistema analógico.
    - Alteración: los sistemas digitales pueden alterarse o manipularse con relativa facilidad con respecto a los analógicos.
* Analógico
  + **Sistema analógico** es cualquier sistema cuyas señales se representan con valores continuos, es decir, que admite números o valores infinitos.
  + Los datos que forman parte de la naturaleza son de origen analógico: la temperatura, la distancia, el sonido, voltaje, imágenes, etc. Si bien todas estas variables se pueden convertir a datos digitales son, originalmente, analógicas.
  + Una señal analógica es un voltaje o corriente que varía suave y continuamente. Una onda senoidal es una señal analógica de una sola frecuencia. Los voltajes de la voz y del video son señales analógicas que varían de acuerdo con el sonido o variaciones de la luz que corresponden a la información que se está transmitiendo.



* + El conversor ADC (Analog-to-Digital Converter – Conversor Analógico Digital) tiene que efectuar los siguientes procesos:
    - Muestreo de la señal analógica
      * Para convertir una señal analógica en digital, el primer paso consiste en realizar un muestreo (sampling) de ésta, o lo que es igual, tomar diferentes muestras de tensiones o voltajes en diferentes puntos de la onda senoidal. La frecuencia a la que se realiza el muestreo se denomina razón, tasa o también frecuencia de muestreo y se mide en kilohertz (kHz). En el caso de una grabación digital de audio, a mayor cantidad de muestras tomadas, mayor calidad y fidelidad tendrá la señal digital resultante.



* + - Cuantización de la señal analógica
      * La cuantización representa el componente de muestreo de las variaciones de valores de tensiones o voltajes tomados en diferentes puntos de la onda sinusoidal, que permite medirlos y asignarles sus correspondientes valores en el sistema numérico decimal, antes de convertir esos valores en sistema numérico binario.



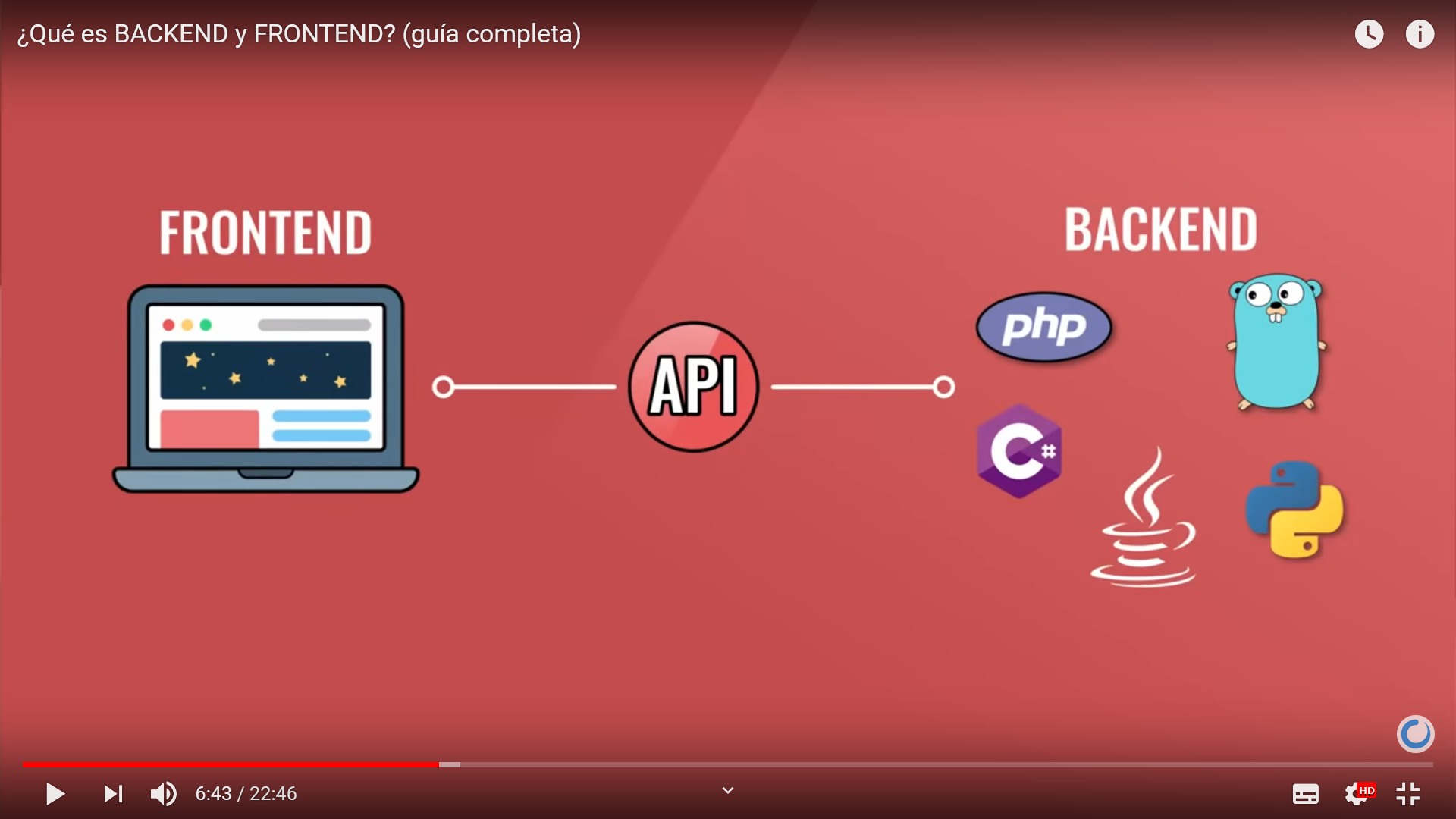
* + - Codificación de la señal en código binario
      * Después de realizada la cuantización, los valores de las tomas de voltajes se representan numéricamente por medio de códigos y estándares previamente establecidos. Lo más común es codificar la señal digital en código numérico binario.

## 1.7. Tarea 25

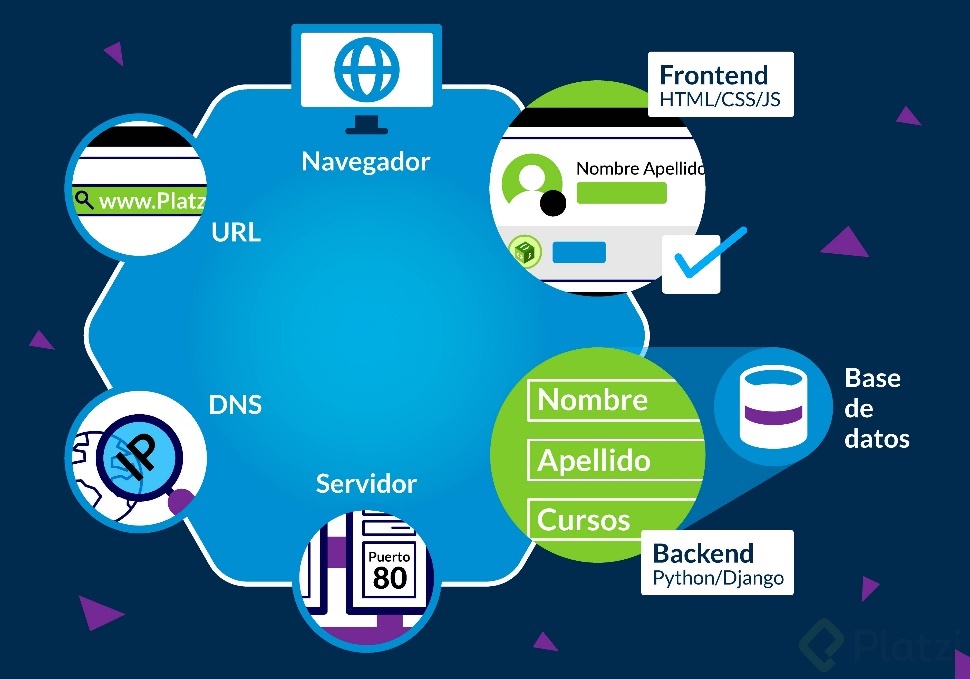
* Arquitectura cliente-servidor (son la clave de internet)



* + Como funciona el proceso desde que entras en Google hasta que aparece en el navegador
    - Cuando hacemos una abrimos una página web nosotros escribimos la dirección, como por ejemplo google.com. Sin embargo, para poder llegar a google.com se necesita saber la IP de Google, que no es otra cosa que una dirección numérica única que identifica de forma única al servidor.
    - Para poder obtener la IP se hace una consulta a un servidor DNS. Esto el usuario no lo ve porque sucede internamente en nuestro sistema.
    - Una vez que nuestro sistema operativo resolvió la **consulta DNS** y sabe cómo encontrar el servidor, nuestro sistema operativo a través del **protocolo TCP/IP** negocia una comunicación con el servidor remoto que aloja la página que queremos ver.
    - Una vez establecida la comunicación, luego mediante el **protocolo HTTPS** y usando una petición tipo de petición GET se solicita la URL en cuestión.
    - En este paso el servidor web debe determinar primero **a que sitio corresponde la petición** (ya que en un mismo servidor pueden alojar varios dominios) y lo hace a través del header recibido observando la URL que se solicitó.
    - Gracias al archivo .htaccess puedes crear URL amigables para ayudar al SEO o simplemente hacer las URL más sencillas, se pueden establecer políticas de seguridad, permisos de acceso o denegación a archivos, filtrados de IP, modificar las cabeceras de respuesta u otros valores de configuración, las opciones son sumamente diversas.
    - A su vez en este punto el servidor debe determinar que se está solicitando, por ejemplo si se trata de una imagen o un archivo HTML el servidor lo leerá del disco SSD y lo enviará al cliente, en cambio si se trata de una página .php o .asp el servidor primero ejecutará el código de dicho script para obtener el contenido que devolverá, que puede ser HTML, un JSON, texto o incluso una imagen o un archivo, a esto es lo que se llama contenido dinámico debido a que no existe en el servidor como tal sino que es generado en el momento para ser mostrado.
    - Una vez que el servidor obtuvo el contenido o datos devolverá una cabecera de respuesta donde por ejemplo especifica el tipo de datos devuelto, en ella indicará si es HTML, un JSON, una imagen o archivo de audio, entre otros, esto facilita al navegador saber que tiene que hacer con ese fichero y cómo debe tratarlo posteriormente.
    - Cada respuesta del servidor va acompañada de un número de identificación que indica que ocurrió. Por ejemplo, si en una petición GET obtenemos el código 200 quiere decir que esta todo correcto y sin errores. Si por el contrario responde con un error 500 será un error interno del servidor, y si recibimos un error 404 es porque la URL no se encontró.
  + Servidor vs Servidor Web
    - Servidor es la máquina física
    - Un servidor web es en realidad un servicio web, es decir, un software que se ejecuta en ese servidor físico o virtual y que despacha tráfico por el protocolo HTTP. En resumen, es un programa informático que se encarga de recibir solicitudes de clientes que solicitan páginas HTML u otro tipo de archivos como imágenes y entregarlos al usuario.
    - Estas solicitudes pueden devolver contenido de dos tipos: **contenido estático** o contenido dinámico.
      * El contenido estático es el contenido que es entregado sin procesar en alguna forma, es decir tal como está almacenado se envía al usuario que lo solicito, como por ejemplo una imagen.
      * El otro tipo de contenido es el llamado **dinámico**, esto significa que el servidor devuelve contenido que no existe como tal en el servidor, sino que es generado en el momento, como por ejemplo la página de resultados de un buscador.
* Frontend y Backend



* + Frontend
    - Es la parte de un programa o dispositivo a la que un usuario puede acceder directamente. Son todas las tecnologías de diseño y desarrollo web que corren en el navegador y que se encargan de la interactividad con los usuarios.
    - Un programador Frontend debe saber de códigos [HTML, CSS y JavaScript]([HTML, CSS y JavaScript](https://platzi.com/cursos/html5-css3/)) para poder usar algunos frameworks o librerías que expanden sus capacidades para crear cualquier tipo de interfaces de usuarios. [React](https://platzi.com/cursos/react/), [Redux](https://platzi.com/cursos/redux/), [Angular](https://platzi.com/cursos/angular/), [Bootstrap, Foundation](https://platzi.com/cursos/fw-frontend/), [LESS](https://platzi.com/cursos/less/), [Sass](https://platzi.com/cursos/sass/), [Stylus](https://platzi.com/cursos/stylus/) y [PostCSS](https://platzi.com/cursos/postcss/) son algunos de ellos.
  + Backend
    - Backend es la capa de acceso a datos de un software o cualquier dispositivo, que no es directamente accesible por los usuarios, además contiene la lógica de la aplicación que maneja dichos datos. El Backend también accede al servidor, que es una aplicación especializada que entiende la forma como el navegador solicita cosas.
    - Algunos de los lenguajes de programación de Backend son [Python](https://platzi.com/cursos/python/), [PHP](https://platzi.com/cursos/php/), [Ruby](https://platzi.com/cursos/ruby/), [C#](https://platzi.com/cursos/c-sharp/) y [Java](https://platzi.com/desarrollo-java/).



## 1.8. Tarea 31

* Proceso ETL (Extract, Transform, Load)
  + Proceso clave para pasar de formatos desestructurados como JSON o XML, con valores nulos, datos errados, caracteres inválidos, registros duplicados y demás problemas a datos totalmente listos para ser analizados.
  + Este proceso tiene tres pasos:
    - Extract
      * Traer los datos del origen
      * El origen de datos o “source” puede contener múltiples fuentes de datos.
    - Transform
      * Transformación de los datos para modelar según lo queramos. Operaciones típicas:
        + Filtrar
        + Eliminar duplicados
        + Calcular datos nuevos
        + …
      * Todo esto se realiza en el área de “staging”, un repositorio temporal
    - Load
      * Una vez los datos están transformados y listos en el área de “staging”, se procede a cargarlos en la base de datos de analítica. Esta base de datos comúnmente es un datawarehouse en donde conviven distintos repositorios de datos.

BIBLIOGRAFIA

* <https://soundgirls.org/entendiendo-los-convertidores-ad-da/>
* <https://es.slideshare.net/ester.gonzalez/arquitectura-cliente-servidor-104004600>
* <https://platzi.com/blog/que-es-frontend-y-backend/>
* <https://www.diferenciador.com/sistema-digital-y-sistema-analogico/>
* <https://bookdown.org/aquintela/EBE/variables-discretas-y-continuas.html>
* <https://www.youtube.com/watch?v=w1K1InJYZh4>
* <https://www.youtube.com/watch?v=-4rUKlNeCEs>
* <https://platzi.com/blog/que-es-un-etl/>
* <https://www.youtube.com/watch?v=u3Le4IFePnQ&t=8s>