# Variables

console.log("test ")

const accountId = 23355;

let accountEmail = "siddharthnaudiyal@gmail.com";

var accountPassword = "Sidhu#daf";

accountCity = "Jaipur";

console.table([accountEmail,accountId,accountPassword,accountCity])

**const** means it cant be changed

**let** and **var difference =** we use let instead of var because in modern day javascript it solves a problem of scope (“{ }”) var was like if u change var it also changes var in whole code irrespective of scope but let only assign value at that scope.

**Don’t use var because of issue in block scope and functional scope.**

* You can assign value in variable without let var const too. It is possible in javascript
* And if you **let**  a variable without any value inside it. Node js would show undefined value

# Datatypes

**Number** 2 to power 53

**Bigint**

**String**

**Boolean** true false

**Null** stand alone value like you wont return zero if you temperature app aint working youll use null cuz zero is value

**Undefined**

**Symbol** unique

**Object**

**Typeof**  it tells you type of data types but type of null is **object**

# Data Conversion

let score = 33

let score2 = "45"

console.log(typeof(score))

let valueInNumber = Number(score2)

console.log(valueInNumber)

console.log(typeof valueInNumber)

**Typeof()** to know the type and **Number()** to convert it into number

* Syntax would be N capital or starts with capital
* When changing number if its not a number in string it will return **NaN**(not a number)
* **Null**  would be converted to **0**
* **Undefined** would be converted to **NaN**
* But they all would be typeof **number**

let isLoggedIn = 1

let isBoolean = Boolean(isLoggedIn)

console.log(isBoolean)

* 1 in Boolean is **true**
* **0** in Boolean is **false**
* **“ “**  is **false**
* **“Siddharth”** is **true**

# Conversion

console.log(null > 0)*//false*

console.log(null == 0)*//false*

console.log(null>=0) *//it converts null to zero (true)*

console.log("2"==2) *//true because it converts string to number auto*

console.log("2"===2)*// falase as it checks its data type too*

# Data types Summary

**Primitive Data types –** strings, number, bigInt, Boolean, null, undefined ,symbol.

They all call by value – that means when make changes they are copy not the actual thing in memory

**Non Primitive (Refrence) -**  Array, Object and Functions

JavaScript is a **dynamically typed language** ✅

That means:

* You **don’t need to declare the type** of a variable when you create it.
* The **type is determined at runtime** (when the code runs), not at compile time.
* A variable can hold different types of values at different times.

const score = 100               *// Number datatype*

const scoreValue = 100.3        *// num*

const isLoggedIn = false        *// boolean*

const outsideTemparture = null  *// null*

let userEmail;                  *// undefined*

const value = Symbol('124')

const value2 = Symbol('124')

console.log(value2===value)  *// false they are unique*

**Datatype of null is Object**

# Stack and heap memory

**Stack(**primitive) as a copy data

let myname = "siddharth"

let anotherName = myname

anotherName = "sid"

console.log(myname)

console.log(anotherName)

it doesn’t change the name of my name

and **Heap (** non primitive) original data

let UserOne = {

    email: "siddharth@gmail.com",

    pass: "whatver"

}

let UserTwo = UserOne;

UserTwo.email = "hellw@gmail.com"

console.log(UserOne.email)  *//hellw@gmail*

console.log(UserTwo.email)  *//hellw#gmail*

it has changed cuz its pointing towards that memory not creating a copy