KQL Statements
SC-200 - DI Thomas Schleich

A. Basics
   a. Operator limit, top
   b. Features of the editor in portal
   c. Operator sort/order
   d. Statement let
   e. Operator project
   f. Operator where
   g. Operator distinct

B. Advanced
   a. Operator extend
   b. Operator summarize
   c. Operator render
   d. Operator union and join
   e. Functions extract and parse
   f. Operator mv_expand, function bag_unpack
   g. Custom Functions

Examples:

Topic	Code

Basics a - d	`DeviceInfo`
	`// Case-sensitivity`
	
	`DeviceInfo`
	`| limit 5`
	`// limit doesn't sort the records.`
	`// Alias for limit: take`
	
	`DeviceInfo`
	`| top 5 by DeviceName desc`
	`// Records sorted first`
	
	`DeviceInfo`
	`| sort by DeviceName asc`
	`DeviceInfo`
	`| sort by DeviceName asc, PublicIP desc`
	`// Alias for sort: order`
	
	`let lmt = 3;`
	`DeviceInfo`
	`| sort by DeviceName asc , Timestamp desc`
	`| limit lmt;`
	`// let creates a variable.`
	
	`let myTable =`
	`DeviceInfo`
	`| limit 10;`
	`myTable`
	`// A variable could also contain a table.`

Topic	Code
Basics e	DeviceInfo | project DeviceName,DeviceType,PublicIP DeviceInfo | project DeviceName,DeviceType,PublicIP | project-away DeviceType DeviceInfo | project-keep Device*,Device*,PublicIP // project-keep has the same result as project but you could use *
Basics f	DeviceInfo | where DeviceName =~ 'Client1' // =~ case-insensitive DeviceInfo | where DeviceName startswith "server" // server* DeviceInfo | where DeviceName endswith ".local" // *.local DeviceInfo | where DeviceName contains "aztrg2112" // *aztrg2112* DeviceInfo | where Timestamp > ago(30m) and Timestamp < ago(15m) DeviceInfo | where startofday(Timestamp) == startofday(now(-1d)) // Yesterday's records – pay attention UTC/Local time format
Basics g	DeviceInfo | project DeviceName,PublicIP | distinct DeviceName,PublicIP
Adv a	AlertEvidence | where EntityType == 'File' | extend FileSizeKB = FileSize / 1024 | project FileName,FileSize,FileSizeKB AlertEvidence | where EntityType == 'File' | extend FileSizeKB = FileSize / 1024, FileSizeMB = FileSize / 1024 / 1024 | project FileName,FileSize,FileSizeKB,FileSizeMB
Adv b	DeviceInfo | project DeviceName | distinct DeviceName DeviceInfo | project DeviceName,PublicIP | distinct DeviceName,PublicIP DeviceInfo | summarize by DeviceName,PublicIP DeviceInfo | summarize count() by DeviceName

Topic	Code
	DeviceInfo | summarize count() by DeviceName,PublicIP DeviceInfo | summarize Qty = count() by DeviceName,PublicIP | sort by DeviceName asc, Qty desc let t = datatable (Hostname:string ,IPAddress:string,Memory:real,CPU:in t) ["Jupiter","1.1.1.1",4096,4, "Venus","2.2.2.2",2048,8, "Mars","3.3.3.3",2048,2]; t | summarize arg_max(CPU,*) by Memory let t = datatable (Hostname:string ,IPAddress:string,Memory:real,CPU:in t) ["Jupiter","1.1.1.1",4096,4, "Venus","2.2.2.2",2048,8, "Mars","3.3.3.3",2048,2]; t | summarize make_list(Hostname) by Memory let t = datatable (Hostname:string ,IPAddress:string,Memory:real,CPU:in t) ["Jupiter","1.1.1.1",4096,4, "Venus","2.2.2.2",2048,8, "Mars","3.3.3.3",2048,2]; t | extend json = pack("Host",Hostname,"RAM",Memory,"CPU",CPU) | summarize make_list(json) by Memory
Adv c	DeviceInfo | summarize Qty = count() by DeviceName | render piechart
Adv d	let myHosts = datatable (Hostname:string, Memory:real, CPU:int, Type:st ring) ["Jupiter",16384,32,"Server", "Mars",8192,8,"Server", "PC01",4096,4,"Client", "PC02",4096,4,"Client", "PC03",2048,2,"Client", "PC04",8192,4,"Client",]; let mySoftware = datatable (Name:string, Version:string, Category:strin g, Hostname:string) ["Adobe Reader", "latest", "Tool","PC03", "Exchange Server 2019","2019","Productivity","Jupiter", "Microsoft 365 Apps", "2019", "Productivity","PC01", "Solitair", "1.0", "Game","PC03", "SQL Server 2019","2019","Productivity","Mars", "Visual Studio Code","latest","Development","PC99"]; // myHosts // | union mySoftware // myHosts // | join mySoftware on Hostname // myHosts // | join kind=fullouter mySoftware on Hostname // myHosts // | join kind=leftouter mySoftware on Hostname

Topic	Code
	// myHosts // | join kind=rightouter mySoftware on Hostname // myHosts // | join kind=rightsemi mySoftware on Hostname let myHosts = datatable (Hostname:string, Memory:real, CPU:int, Type:st ring) ["Jupiter",16384,32,"Server", "Mars",8192,8,"Server", "PC01",4096,4,"Client", "PC02",4096,4,"Client", "PC03",2048,2,"Client", "PC04",8192,4,"Client",]; let mySoftware = datatable (Name:string, Version:string, Category:strin g, Computer:string) ["Adobe Reader", "latest", "Tool","PC03", "Exchange Server 2019","2019","Productivity","Jupiter", "Microsoft 365 Apps", "2019", "Productivity","PC01", "Solitair", "1.0", "Game","PC03", "SQL Server 2019","2019","Productivity","Mars", "Visual Studio Code","latest","Development","PC99"]; let myInventory = myHosts | join mySoftware on $left.Hostname == $right.Computer | project-away Computer; myInventory
Adv e	DeviceInfo | extend hostname = extract(@"(^[a-z|A-Z|0-9]+)",0,DeviceName), domain = extract(@"(^[a-z|A-Z|0-9]+)\.(.+)$",2,DeviceName) | project DeviceName,hostname,domain let myPlanets = datatable (Item:int, Statement:string) [1, "The planet Mercury has a circumference of 15330 km and a mass of 0.33 E+24 kg.", 2, "The planet Venus has a circumference of 38023 km and a mass of 4.87 E+24 kg.", 3, "The planet Earth has a circumference of 40075 km and a mass of 5.97 E+24 kg.", 4, "The planet Mars has a circumference of 21344 km and a mass of 0.64 E+24 kg."]; myPlanets | parse Statement with * "planet " planet:string " has a circumference of " circum:long " km and a mass of " mass:real " E" * | project Item,planet,circum,mass
Adv f	let myHosts = datatable (Hostname:string, Hardware:dynamic , Software:dynamic , Location:string , Devices:dynamic) ["Jupiter", dynamic({"RAM":8192,"CPU":12,"DiskCapacityGB":512,"DiskCount":6,"DiskVe ndor":"WD"}), dynamic([{"Product":"Exchange","Vendor":"Microsoft","Version":"2019","B uiltIn":false},{"Product":"FileServer","Vendor":"Microsoft","Version":" 2022","BuiltIn":true}]), "Graz", dynamic(["Mouse","FIDO2-USB"]), "Saturn", dynamic({"RAM":8192,"CPU":8,"DiskCapacityGB":768,"DiskCount":12,"DiskVe ndor":"WD"}), dynamic([{"Product":"SharePoint","Vendor":"Microsoft","Version":"2019", "BuiltIn":false},{"Product":"DC","Vendor":"Microsoft","Version":"2022", "BuiltIn":true},{"Product":"DNS","Vendor":"Microsoft","Version":"2022", "BuiltIn":true}]), "Graz", dynamic(["Mouse","iPhone"]),

Topic	Code
	"Neptune", dynamic({"RAM":4096,"CPU":2,"DiskCapacityGB":256,"DiskCount":4,"DiskVendor":"IBM"}), dynamic([{"Product":"S/4HANA","Vendor":"SAP","Version":"2021","BuiltIn":false},{"Product":"PrintServer","Vendor":"Microsoft","Version":"2022","BuiltIn":true}]), "Linz", dynamic(["Pen","iPhone"]), "Uranos", dynamic({"RAM":6144,"CPU":8,"DiskCapacityGB":512,"DiskCount":8,"DiskVendor":"IBM"}), dynamic([{"Product":"Exchange","Vendor":"Microsoft","Version":"2019","BuiltIn":false},{"Product":"DC","Vendor":"Microsoft","Version":"2022","BuiltIn":true},{"Product":"DNS","Vendor":"Microsoft","Version":"2022","BuiltIn":true}]), "Linz", dynamic(["Mouse","FIDO2-USB"])]; myHosts // | mv-expand Devices | mv-expand Software | evaluate bag_unpack(Software)

Adv g	let host = extract(@"(^[a-z|A-Z|0-9]+)",0,FQDN);
	let domain = extract(@"(^[a-z|A-Z|0-9]+)\.(.+)$",2,FQDN);
	bag_pack("Hostname",host,"Domain",domain)
	
	// Save this code as *Function* and declare the string parameter **FQDN**.