

Intelligence = Cloud + Data + Automation

9i, 10g, 11g 12cc(10DBA) --> Auto Heal---> 18c(3/4 DBA)

Cloud:

- > Set of Services
- > Services:
 - > Storage
 - > Computer
 - a. ServerBase
 - 1. Dedicated
 - 2. Virtualize(Shared)
 - 3. Containers
 - b. Serverless
 - 1. Microservices
 - > Network
 - Security

Web Framework	Django
Database Server	SQL Server
Runtime	VM --> Serverless
VCS	Github
Editor	Online/Offline VSCode
Package Availability	1. VM(Terminal) 2. SSH Tunnel

SQL

- Query Language
- Standard (ANSI SQL)
 - Close Source :
 - MS SQL Server(Paid)
 - Open Source :
 - MySQL Server(Community)
- MS SQL Server
 - Implementation of
 - Standard SQL
 - RDBMS
 - Package for OORDBMS(BI)---> DWH(ETL) --> BigData I
 - SSI(Integration)S, SSR(Reporting)S, SSA(Analysis)S
 - Tools
 - CLI - SQL Trace
 - GUI - SQL Profiler
 - DB IDE - Data Studio
 - SQL Client (GUI) - Sql Server Management Studio
 - SQL Client (CLI) - SQLCMD

Tools

- CLI - SQL Trace
- GUI - SQL Profiler
- DB IDE - Data Studio
- SQL Client (GUI) - Sql Server Management Studio, DBeaver
- SQL Client (CLI) - SQLCMD

FMS(File) --> DBMS

RDBMS --> OORDBMS, NDBMS, HDBMS

Data

1. Traditional Data
2. Big Data
 - a. Hadoop --> PySpark --> Streaming ---> Data Lake
3. New Data

Driver

- Driver is basically software which helps to understand system(hw/sw)
 - Infinite program (program/service/daemon)

Azure sql

- Server → infinite program to bring data files
- Database → set of objects (table, view, stored procedure, functions, packages etc)
 - Master(Simple Data Entry Table)
 - Helps to Perform transactions
 - Product
 - Category
 - Transactional(Relational Complex)
 - Stock
 - Sales
 - Purchase
 - Temp Table
 - CTE(Common Table Expression)

➤ Client

- Web - Web Query
- GUI - Dbaever
- CLI - SQLCMD

Virtualization(ms azure) Containerization(Docker)

Image-File(local/Remote)

Container-Process/Running image

(Archive registry/container registry

Market place(Remote img)

-Hub.Docker.com

-Microsoft-→ACR-Azure Container Registry

-GCR,ECR etc

Shops(repository)

Docker pull mujahed/welcome1:tagname

ShopName/Product Name:version

1. Check Subscription
2. Create Resource Group
3. Order VM
 - a. Configuration: 4CPU, 16GB
 - b. UserName: azureuser
 - c. Password: Localhost@1234567

-
4. Connect VM
 - a. Azure CLI
 - i. \$ az ssh vm --resource-group rg-24jan-pentaho --vm-name vm-24jan-pentaho --subscription 675613e4-01ce-4876-99a1-37bb6cffdd3d
 - ii. \$ az ssh vm --resource-group r1 --vm-name v1 --subscription s1
 - b. SSH CLI
 - c. GUI Tools
 - i. MobaXterm
 - ii. Putty
 5. Install Docker
-

III. TUTORIAL

5. Install Docker

-
- a. \$ sudo su
 - b. # apt update
 - c. # apt install docker.io -y
-

6. Create Application Deployment

-
- a. Create Docker Network
 - i. # docker network create incedo-nw
 - ii. # docker network ls
 - b. Pull Remote Image From CR(hub.docker.com)
 - i. # docker images
 - ii. # docker pull nginx:latest
 - c. Local Image(nginx:latest) --> Container [detachable mode]
 - i. # docker run --rm --net nw --name ws -d nginx:latest
 - ii. # export CID=1ed32814ee6c
 - iii. # docker inspect \$CID | grep IPAddress
 - iv. # export WSIP=172.18.0.2
 - d. Pull Remote Image--> Local Image --> Container
 - i. # docker run --net nw --it busybox sh
 - e. Test Web Server From 2nd Container
 - e. Test Web Server From 2nd Container
 - i. # wget -q -O - http://ContainerName:80
 - ii. # wget -q -O - http://IPAddress:80
 - iii. wget -q -O - http://IPAddress
 - iv. wget -q -O - IPAddress
 - v. wget -q -O - ContainerName
 - f. Delete All Containers & Images
 - i. # docker rm \$(docker ps -aq) --force && docker rmi \$(docker images -q) --force
-

Inedo SSH using Azure CLI - Microsoft Edge

portal.azure.com/#@incedoin.onmicrosoft.com/resource/subscriptions/085dc004-4699-4129-a50d-65fe4a17edac/resourceGroups/r24janyash

Microsoft Azure Search resources, services, and docs (G+)

Home > vm24janyash

vm24janyash | Connect Virtual machine

Search Refresh Troubleshoot More Options Fee

Connecting using Public IP address | 52.146.95.10

Admin username

Bash

```
--header STR   Add STR (of form 'header: value') to headers
--post-data STR Send STR using POST method
--post-file FILE Send FILE using POST method
--no-check-certificate Don't validate the server's certificate
-c             Continue retrieval of aborted transfer
-q             Quiet
-P DIR        Save to DIR (default .)
-S             Show server response
-T SEC        Network read timeout is SEC seconds
-O FILE       Save to FILE ('-' for stdout)
-o LOGFILE    Log messages to FILE
-U STR        Use STR for User-Agent header
-Y on/off     Use proxy
/ #
```

Type here to search inedo

29°C ENG 15:43 24-01-2024

SSH using Azure CLI

Connect from the Azure portal

Connect from your local machine

1 **Configure prerequisites for SSH using Azure CLI**

Azure needs to configure some features in order to connect to the VM.

Prerequisites configured

System assigned managed identity

Azure will configure a system-assigned managed identity in order to enable the Azure AD login extension. [Learn more](#)

Microsoft Azure Search resources, services, and docs (G+)

Home > vm24janyash

vm24janyash | Connect Virtual machine

Search Refresh Troubleshoot More Options Fee

Connecting using Public IP address | 52.146.95.10

Admin username

Bash

```
9ad6333ebc9: Pull complete
Digest: sha256:6d9ac9237a84afe1516540f40a0fafdc86859b2141954b4d643af7066d598b74
Status: Downloaded newer image for busybox:latest
docker: Error response from daemon: network nw not found.
ERRO[0000] error waiting for container:
root@vm24janyash:/home/yashwardhan.singh# docker run --net inedo-nw -it busybox sh
/ # wget -q -O - http://172.18.0.2:80
BusyBox v1.36.1 (2024-01-17 21:57:33 UTC) multi-call binary.

Usage: wget [-cqS] [--spider] [-O FILE] [-o LOGFILE] [--header STR]
           [-p post-data STR | --post-file FILE] [-Y on/off]
           [-n --no-check-certificate] [-P DIR] [-U AGENT] [-T SEC] URL...
Retrieve files via HTTP or FTP
```

Type here to search inedo

29°C ENG 15:43 24-01-2024

SSH using Azure CLI

Connect from the Azure portal

1 **Configure prerequisites for SSH using Azure CLI**

Azure needs to configure some features in order to connect to the VM.

- Prerequisites configured**
- System assigned managed identity**

root@vm24janyash:~# docker inspect \$CID|grep IPAddress

```

    "MacAddress": "02:42:ac:12:00:02",
    "DriverOpts": null
}
}
]
root@vm24janyash:/home/yashwardhan.singh# docker inspect $CID|grep IPAddress
    "SecondaryIPAddresses": null,
    "IPAddress": "",
    "IPAddress": "172.18.0.2",
root@vm24janyash:/home/yashwardhan.singh# export WSIP=172.18.0.2
root@vm24janyash:/home/yashwardhan.singh# docker run --net nw -it busybox sh
Unable to find image 'busybox:latest' locally

```

SSH using Azure CLI

Connect from the Azure portal

1 **Configure prerequisites for SSH using Azure CLI**

Azure needs to configure some features in order to connect to the VM.

- Prerequisites configured**
- System assigned managed identity**

root@vm24janyash:~# docker images

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
nginx	latest	a8758716bb6a	3 months ago	187MB

```

root@vm24janyash:/home/yashwardhan.singh# docker run --rm --net incedo-nw --name ws -d nginx:latest
c9662dfe958a465c62b408fab2c64d09fcfc8c95ff501fa2489812c4f9a6e1
root@vm24janyash:/home/yashwardhan.singh# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
c9662dfe958a nginx:latest "/docker-entrypoint..." 12 seconds ago Up 11 seconds 80/tcp ws
264ece7a4974 nginx:latest "/docker-entrypoint..." 7 minutes ago Up 7 minutes 80/tcp objective_tu
34279748dd99 nginx "/docker-entrypoint..." 15 minutes ago Up 15 minutes 80/tcp lucid_boyd
root@vm24janyash:/home/yashwardhan.singh# export CID=c9662dfe958a

```

Inedo SSH using Azure CLI - Microsoft Edge

portal.azure.com/#@incedoin.onmicrosoft.com/resource/subscriptions/085dc004-4699-4129-a50d-65fe4a17edac/resourceGroups/rg24janyash

Microsoft Azure Search resources, services, and docs (G+)

Home > vm24janyash

vm24janyash | Connect

Virtual machine

Search Refresh Troubleshoot More Options Fee

Connecting using Public IP address | 52.146.95.10

Admin username

Bash

```
c9662dfef958a  nginx:latest  "/docker-entrypoint..."  11 minutes ago  Up 11 minutes  80/tcp   ws
264ece7a4974  nginx:latest  "/docker-entrypoint..."  18 minutes ago  Up 18 minutes  80/tcp   objective_tu
34279748dd99  nginx  "/docker-entrypoint..."  27 minutes ago  Up 27 minutes  80/tcp   lucid_boyd
root@vm24janyash:/home/yashwardhan.singh# docker run --net inedo-nw -it busybox sh
/ # wget -q -O -c9662dfef958a
BusyBox v1.36.1 (2024-01-17 21:57:33 UTC) multi-call binary.

Usage: wget [-cqS] [--spider] [-O FILE] [-o LOGFILE] [--header STR]
          [-post-data STR | --post-file FILE] [-Y on/off]
          [-no-check-certificate] [-P DIR] [-U AGENT] [-T SEC] URL...
Retrieve files via HTTP or FTP
--spider      Only check URL existence: $? is 0 if exists
```

Type here to search inedo

SSH using Azure CLI

Connect from the Azure portal

Connect from your local machine

1 Configure prerequisites for SSH using Azure CLI

Azure needs to configure some features in order to connect to the VM.

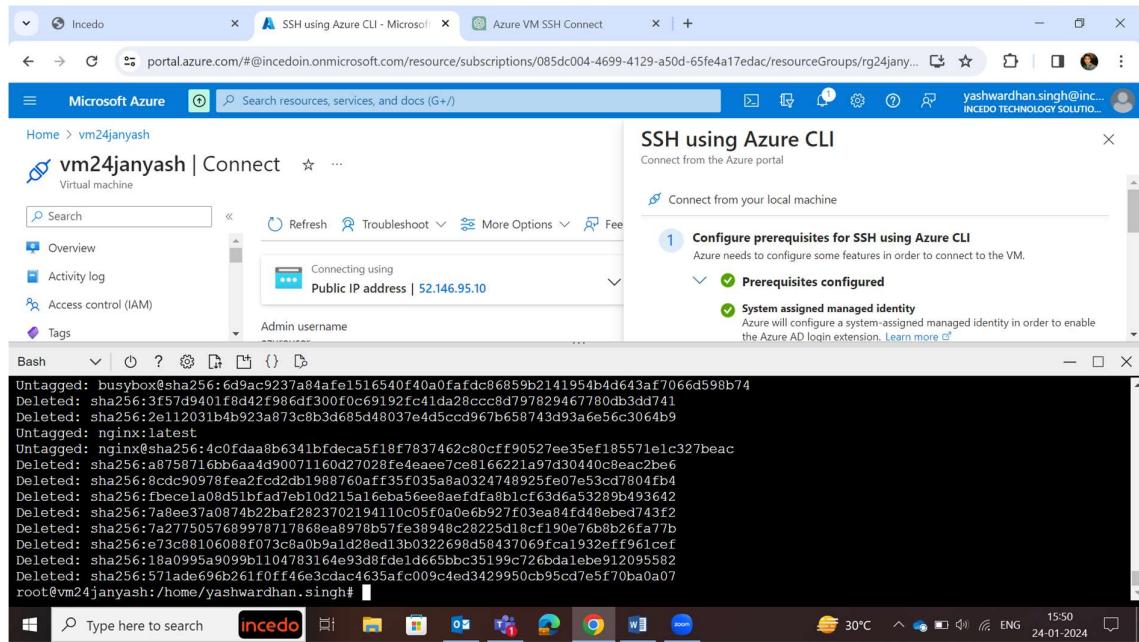
Prerequisites configured

System assigned managed identity

Azure will configure a system-assigned managed identity in order to enable the Azure AD login extension. [Learn more](#)

```
/ # exit
root@vm24janyash:/home/yashwardhan.singh# docker rm $(docker ps -aq) --force && docker rmi $(docker images -q) --force
658e10609474
8b787da71870
8c60db8a7f75
c9662dfef958a
264ece7a4974
34279748dd99
Untagged: busybox:latest
Untagged: busybox@sha256:6d9ac9237a84afe1516540f40a0fafdc86859b2141954b4d643af7066d598b74
Deleted: sha256:3f57d9401fb42f986df300fc69192fc41da28ccc8d797829467780db3dd741
Deleted: sha256:2e112031b4b923a873cb83d685d48037e4d5cc967b658743d93a6e56c3064b9
Untagged: nginx:latest
Untagged: nginx@sha256:4cfdaa8b6341bfdeca5f18f7837462c80cff90527ee35ef185571e1c327beac
```

Type here to search inedo



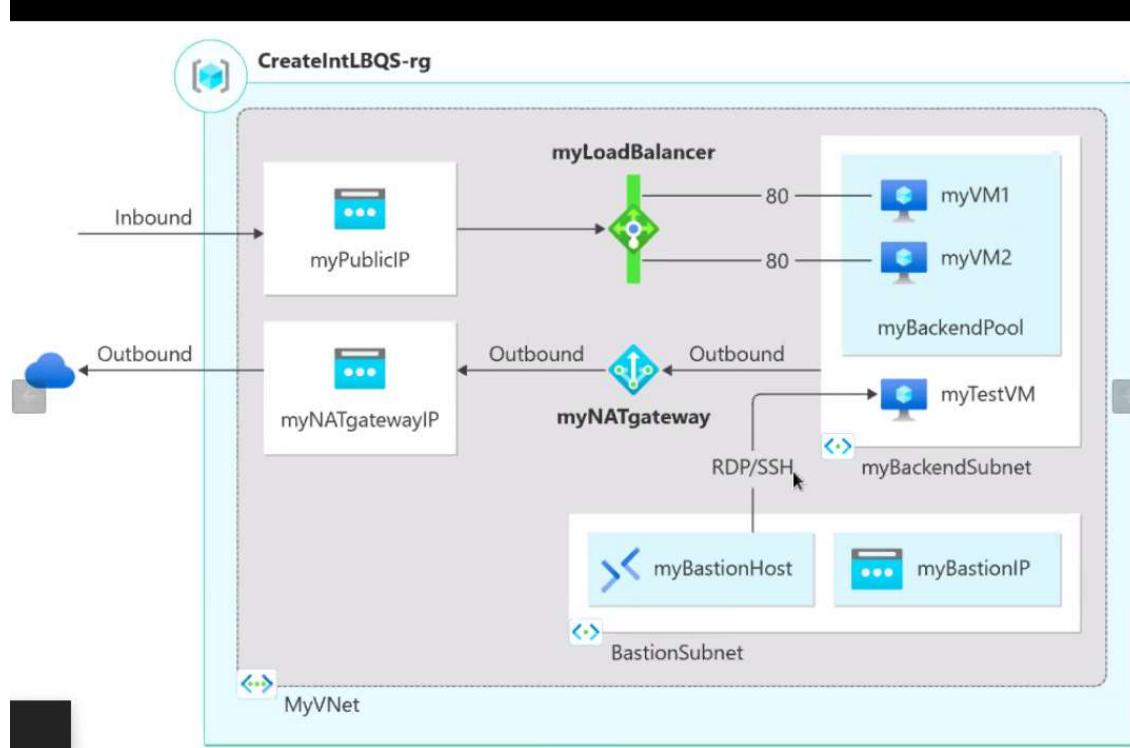
29 jan 2024

Typical Application:

App1 → WebServer → VM(OS) → IP

Why Azure Load Balancer:

- > Load Balance internal/external traffic(Request/Response) to Azure VM
- > Increase availability distribute resources within + across = Zones
- > Configure outbound connectivity to LB resources
 - Internet → Incoming traffic → VM =Inbound
 - Internet ← Outgoing traffic ← VM =Outbound
- > Use health probes to monitor LB resources(VMs)
- > Port Forwarding



NSG:Network security group

HP: health probe

NIC: network interface card

- 1) Subscription → Resource Group
- 2) VNet → myPubIP
- 3) LB → HP → LB Route
- 4) NSG → NSG Rule
- 5) Bastion Host
 - PubIP - MyBastionIP
 - Subnet - |

Subnet - AzBastionSubnet

6) Backend Subnet:

Create NIC with name as MyNicVM1, MyNicVM2

6) Backend Subnet:

Create NIC with name as MyNicVM1, MyNicVM2

VM1, VM2

LB ← VM1, VM2 ← MyNicVM1, MyNicVM2

7) NAT Gateway

MyNATGWIP

MyNATGW → BackendSubnet

8) WebServer(IIS) ← HelloWorld ← VM1, VM2

9) Test

10) Delete

Full CIDR Block: 10.1.0.0/26

Network Type: 10.1.0.

IPV4 : X.X.X.X = 8.8.8.8 = 32 - 26 = 2^6 = 64

Start IP Address: 10.1.0.0

End IP Address : 10.1.0.63

Full CIDR Block: 10.1.0.0/28

Network Type: 10.1.0.

IPV4 : X.X.X.X = 8.8.8.8 = 32 - 28 = 2^4 = 16

Start IP Address: 10.1.0.0

End IP Address : 10.1.0.15

```
az group create --name CreatePubLBQS-rg \
```

```
> --location eastus
```

```
yashwardhan [ ~ ]$ az group create --name CreatePubLBQS-rg \
> --location eastus
{
  "id": "/subscriptions/085dc004-4699-4129-a50d-65fe4a17edac/resourceGroups/CreatePubLBQS-rg",
  "location": "eastus",
  "managedBy": null,
  "name": "CreatePubLBQS-rg",
  "properties": {
    "provisioningState": "Succeeded"
  },
  "tags": null,
  "type": "Microsoft.Resources/resourceGroups"
}
```

```
az network vnet create \
--resource-group CreatePubLBQS-rg \
--location eastus \
--name myVNet \
--address-prefixes 10.1.0.0/16 \
--subnet-name myBackendSubnet \
--subnet-prefixes 10.1.0.0/24
```

```
yashwardhan [ ~ ]$ az network vnet create \
--resource-group CreatePubLBQS-rg \
--location eastus \
--name myVNet \
--address-prefixes 10.1.0.0/16 \
--subnet-name myBackendSubnet \
--subnet-prefixes 10.1.0.0/24
{
  "newVNet": {
    "addressSpace": {
      "addressPrefixes": [
        "10.1.0.0/16"
      ]
    },
    "enableDdosProtection": false,
    "etag": "W/\\"bf51b86e-6551-44bd-9bea-5d2fd78ce73a\\"",
    "id": "/subscriptions/085dc004-4699-4129-a50d-65fe4a17edac/resourceGroups/CreatePubLBQS-rg/providers/Microsoft.Network/virtualNetworks/myVNet",
    "location": "eastus",
    "name": "myVNet",
    "provisioningState": "Succeeded",
    "resourceGroup": "CreatePubLBQS-rg",
    "resourceGuid": "5f24cdc9-8ffe-404d-bd02-970e0b28164e",
    "subnets": [
      {
        "addressPrefix": "10.1.0.0/24",
        "delegations": []
      }
    ]
  }
}
```

```
az network public-ip create --resource-group CreatePubLBQS-rg --name myPublicIP --
sku Standard --zone 1
```

```
yashwardhan [ ~ ]$ az network public-ip create --resource-group CreatePubLBQS-rg --name myPublicIP --sku Standard --zone 1
{
  "publicIp": {
    "ddosSettings": {
      "protectionMode": "VirtualNetworkInherited"
    },
    "etag": "W/"c3f2d86f-d83a-442f-8e3e-f92c439cbfc4"",
    "id": "/subscriptions/085dc004-4699-4129-a50d-65fe4a17edac/resourceGroups/CreatePubLBQS-rg/providers/Microsoft.Network/publicIPAddresses/myPublicIP",
    "idleTimeoutInMinutes": 4,
    "ipAddress": "172.178.88.59",
    "ipTags": [],
    "location": "eastus",
    "name": "myPublicIP",
    "provisioningState": "Succeeded",
    "publicIPAddressVersion": "IPv4",
    "publicIPAllocationMethod": "Static",
    "resourceGroup": "CreatePubLBQS-rg",
    "resourceGuid": "afaae6b7-9c4c-42ad-95ce-54dbf4b4c456",
    "sku": {
      "name": "Standard",
      "tier": "Regional"
    },
    "type": "Microsoft.Network/publicIPAddresses",
    "zones": [
      "1"
    ]
  }
}
```

```
az network lb create \
--resource-group CreatePubLBQS-rg \
--name myLoadBalancer \
--sku Standard \
--public-ip-address myPublicIP \
--frontend-ip-name myFrontEnd \
--backend-pool-name myBackEndPool
```

```
yashwardhan [ ~ ]$ az network lb create \
  --resource-group CreatePubLBQS-rg \
  --name myLoadBalancer \
  --sku Standard \
  --public-ip-address myPublicIP \
  --frontend-ip-name myFrontEnd \
  --backend-pool-name myBackEndPool
{
  "loadBalancer": {
    "backendAddressPools": [
      {
        "etag": "W/"flcc477e-0412-44a1-b5f1-34b0a124f57b"",
        "id": "/subscriptions/085dc004-4699-4129-a50d-65fe4a17edac/resourceGroups/CreatePubLBQS-rg/providers/Microsoft.Network/loadBalancers/myLoadBalancer/backendAddressPools/myBackEndPool",
        "name": "myBackEndPool",
        "properties": {
          "loadBalancerBackendAddresses": [],
          "provisioningState": "Succeeded"
        },
        "resourceGroup": "CreatePubLBQS-rg",
        "type": "Microsoft.Network/loadBalancers/backendAddressPools"
      }
    ],
    "frontendIPConfigurations": [
      {
        "etag": "W/"flcc477e-0412-44a1-b5f1-34b0a124f57b"",
        "id": "/subscriptions/085dc004-4699-4129-a50d-65fe4a17edac/resourceGroups/CreatePubLBQS-rg/providers/Microsoft.Network/loadBalancers/myLoadBalancer/frontendIPConfigurations/myFrontEnd",
        "name": "myFrontEnd",
        "properties": {
          "privateIPAllocationMethod": "Dynamic",
          "provisioningState": "Succeeded"
        }
      }
    ]
  }
}
```

Showing 1 to 3 of 3 records. Show hidden types ⓘ

Name ↑↓	Type ↑↓	Location ↑↓
<input type="checkbox"/> myLoadBalancer	Load balancer	East US
<input type="checkbox"/> myPublicIP	Public IP address	East US
<input type="checkbox"/> myVNet	Virtual network	East US

```
az network lb probe create \
--resource-group CreatePubLBQS-rg \
--lb-name myLoadBalancer \
--name myHealthProbe \
--protocol tcp \
--port 80
```

```
az network lb rule create \
--resource-group CreatePubLBQS-rg \
--lb-name myLoadBalancer \
--name myHTTPRule \
--protocol tcp --frontend-port 80 \
--backend-port 80 \
--frontend-ip-name myFrontEnd \
--backend-pool-name myBackEndPool \
--probe-name myHealthProbe \
--disable-outbound-snat true \
--idle-timeout 15 --enable-tcp-reset true
```

```
az network nsg create \
--resource-group CreatePubLBQS-rg \
--name myNSG
```

```
az network nsg rule create \
--resource-group CreatePubLBQS-rg \
```

```
--nsg-name myNSG \
--name myNSGRuleHTTP \
--protocol '*' --direction inbound \
--source-address-prefix '*' \
--source-port-range '*' \
--destination-address-prefix '*' \
--destination-port-range 80 \
--access allow --priority 200
```

```
az network public-ip create \
--resource-group CreatePubLBQS-rg \
--name myBastionIP \
--sku Standard --zone 1 2 3
```

```
az network vnet subnet create \
--resource-group CreatePubLBQS-rg \
--name AzureBastionSubnet \
--vnet-name myVNet \
--address-prefixes 10.1.1.0/27
```

```
az network bastion create \
--resource-group CreatePubLBQS-rg \
--name myBastionHost \
--public-ip-address myBastionIP \
--vnet-name myVNet \
--location eastus
```

```
array=(myNicVM1 myNicVM2)
for vmnic in "${array[@]}"
do
    az network nic create \
        --resource-group CreatePubLBQS-rg \
        --name $vmnic \
        --vnet-name myVNet \
        --subnet myBackEndSubnet \
        --network-security-group myNSG
done
```

```
az vm create \
--resource-group CreatePubLBQS-rg \
--name myVM1 \
--nics myNicVM1 \
--image win2019datacenter \
--admin-username azureuser \
--zone 1 --no-wait
```

localhost@1234567

```
az vm create \
--resource-group CreatePubLBQS-rg \
--name myVM2 \
--nics myNicVM2 \
--image win2019datacenter \
--admin-username azureuser \
--zone 2 --no-wait
```

```
--image win2019datacenter \
--admin-username azureuser \
--zone 1 --no-wait

Admin Password:
Confirm Admin Password:
yashwardhan [ ~ ]$ az vm create \
--resource-group CreatePubLBQS-rg \
--name myVM2 \
--nics myNicVM2 \
--image win2019datacenter \
--admin-username azureuser \
--zone 2 --no-wait
Admin Password:
Confirm Admin Password:
The password length must be between 12 and 123. Password must have the 3 of the following: 1 lower case character, 1 upper case character, 1 number and 1 special character.
yashwardhan [ ~ ]$ az vm create \
--resource-group CreatePubLBQS-rg \
--name myVM2 \
--nics myNicVM2 \
--image win2019datacenter \
--admin-username azureuser \
--zone 2 --no-wait
Admin Password:
Confirm Admin Password:
yashwardhan [ ~ ]$
```

```
array=(myNicVM1 myNicVM2)
for vmnic in "${array[@]}"
do
    az network nic ip-config address-pool add \
        --address-pool myBackendPool \
        --ip-config-name ipconfig1 \
        --nic-name $vmnic \
```

```
--resource-group CreatePubLBQS-rg \
--lb-name myLoadBalancer
done
```

```
az network public-ip create \
--resource-group CreatePubLBQS-rg \
--name myNATgatewayIP \
--sku Standard \
--zone 1 2 3
```

```
az network nat gateway create \
--resource-group CreatePubLBQS-rg \
--name myNATgateway \
--public-ip-addresses myNATgatewayIP \
--idle-timeout 10
```

```
az network vnet subnet update \
--resource-group CreatePubLBQS-rg \
--vnet-name myVNet \
--name myBackendSubnet \
--nat-gateway myNATgateway
```

```
array=(myVM1 myVM2)
for vm in "${array[@]}"
do
    az vm extension set \
        --publisher Microsoft.Compute \
        --version 1.8 --name CustomScriptExtension \
        --vm-name $vm --resource-group CreatePubLBQS-rg \
        --settings '{"commandToExecute":"powershell Add-WindowsFeature Web-Server; powershell Add-Content -Path \"C:\\inetpub\\wwwroot\\Default.htm\" -Value $($env:computername)"}'
done
```

```

yashwardhan [ ~ ]$ array=(myVM1 myVM2)
for vm in "${array[@]}"
do
    az vm extension set \
        --publisher Microsoft.Compute \
        --version 1.8 --name CustomScriptExtension \
        --vm-name $vm --resource-group CreatePubLBQS-rg \
        --settings '{"commandToExecute":"powershell Add-WindowsFeature Web-Server; powershell Add-Content -Path \"C:\\inetpub\\wwwroot\\Default.htm\" -Value $($env:computername)"}'
done
{
    "autoUpgradeMinorVersion": true,
    "enableAutomaticUpgrade": null,
    "forceUpdateTag": null,
    "id": "/subscriptions/085dc004-4699-4129-a50d-65fe4a17edac/resourceGroups/CreatePubLBQS-rg/providers/Microsoft.Compute/virtualMachines/myVM1/extensions/CustomScriptExtension",
    "instanceView": null,
    "location": "eastus",
    "name": "CustomScriptExtension",
    "protectedSettings": null,
    "protectedSettingsFromKeyVault": null,
    "provisionAfterExtensions": null,
    "provisioningState": "Succeeded",
    "publisher": "Microsoft.Compute",
    "resourceGroup": "CreatePubLBQS-rg",
    "settings": {
        "commandToExecute": "powershell Add-WindowsFeature Web-Server; powershell Add-Content -Path \"C:\\inetpub\\wwwroot\\Default.htm\" -Value $($env:computername)"}}

```

```

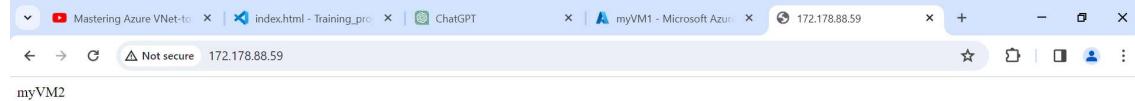
az network public-ip show \
    --resource-group CreatePubLBQS-rg \
    --name myPublicIP \
    --query ipAddress \
    --output tsv

```

```

    "typePropertiesType": "CustomScriptExtension"
}
yashwardhan [ ~ ]$ az network public-ip show \
    --resource-group CreatePubLBQS-rg \
    --name myPublicIP \
    --query ipAddress \
    --output tsv
Command group 'az network' is in preview and under development. Reference and support levels: https://aka.ms/CLI_refstatus
172.178.88.59

```



<input type="checkbox"/>  myBastionHost	Bastion	East US
<input type="checkbox"/>  myBastionIP	Public IP address	East US
<input type="checkbox"/>  myLoadBalancer	Load balancer	East US
<input type="checkbox"/>  myNATgateway	NAT gateway	East US
<input type="checkbox"/>  myNATgatewayIP	Public IP address	East US
<input type="checkbox"/>  myNicVM1	Network Interface	East US
<input type="checkbox"/>  mvNicVM2	Network Interface	East US

<input type="checkbox"/>  myNicVM2	Network Interface	East US
<input type="checkbox"/>  myNSG	Network security group	East US
<input type="checkbox"/>  myPublicIP	Public IP address	East US
<input type="checkbox"/>  myVM1_OsDisk_1_14aaaf37f4e3a481d9b55f6c8407e27b9	Disk	East US
<input type="checkbox"/>  myVM2	Virtual machine	East US
<input type="checkbox"/>  myVM2_OsDisk_1_150d2818af664e5ab7f2fca7c3f7d426	Disk	East US
<input type="checkbox"/>  myVNet	Virtual network	East US

*Sast and Dast

For agile paid tool jira

Trello tool which maintain ticket last date efforts etc → category of ticketing tools

<https://drive.google.com/file/d/1mODaphOZ18BEUqaM4MqKjth5s49OijJP/view?usp=sharing>
<https://drive.google.com/file/d/133JEnborTj8mX--MUaN4K2EwhZUAk9Vi/view?usp=sharing>
https://drive.google.com/file/d/1Kifc2gBbAfJnmcQwHI5pZPAxx_fdj5BF/view?usp=sharing
<https://drive.google.com/file/d/1qWnwLNAV7CUPCsEEMJ9QYrzTZlyAetgA/view?usp=sharing>
<https://drive.google.com/file/d/1Y2ZGkXO-VlhiO0ILbnYkUIEMLUEqOxgG/view?usp=sharing>
<https://drive.google.com/file/d/1sixq4TD6dJrefom6ci4RuePaNXDIM6ZI/view?usp=sharing>
<https://drive.google.com/file/d/1yIVOR86mZSu0n6w-wKnFmrJksvb4HV2M/view?usp=sharing>
https://drive.google.com/file/d/1voVZY8N1tnQLnoGZEJxtUYjas6k_v2QY/view?usp=sharing
<https://drive.google.com/file/d/1aoFq3HgOdXhunj9QBdSH3a3a7rRzCwv2/view?usp=sharing>

```

Agile
    Documentation: Confluence, Github pages
    Ticketing Tool: Trello, Jira, Github Issue
    Chat: Slack, Microsoft Team

Designing
    Graphic: Corel, Photoshop, AI
    Web/Mobile App: AdobeXD, Figma
    Database: ER Diagram, Draw.io
    Software: UML

Development
    Distributed VCS: Github, GitLab Repo, Bitbucket
    Testing:
        Automated Testing: Selenium, JUnit, TestNG,
        Continues Testing: Jenkins, Github
    Continues Integration: Jenkins, Github Action(Infra Automation), GitLab
    Continues Deployment/Configuration/Delivery/Release:
        Infra. Provisioning: Terraform
        Continues Deployment: Ansible/Jenkins/Docker
        Configuration Mgt: Ansible

```

```

git clone https://github.com/NubeEra-Projects/LnT.git

1) Push all Changes(LnT) in LXP project
    1.1 - Push LnT in Github
        Gaurav
        Yuvraj
        Deepak
    1.2 - LXP.zip → Push in LnT
        Remaining Leaners
    1.3 - Testing
        |
2) Web Designining(UI Changes) - CSS, Bootstrap
3) Sqlite → MS SQL Server
4) Django Web Server → NGINX(Web/ProxyServer) & Gunicorn(WSGI)
5) Documentations
    Architecture
    Modules
    Database Design

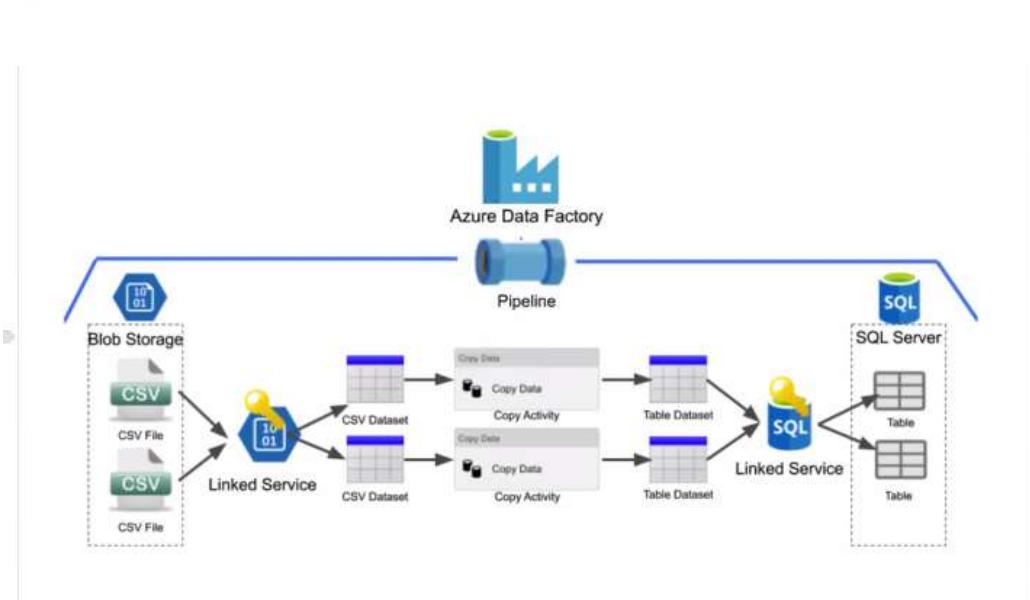
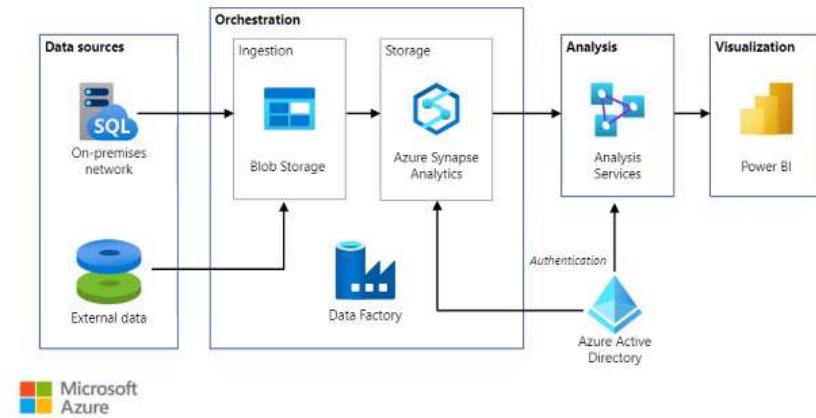
```

02/02/2023

Data factory

Code free ETL as a service

1. Extract the data (Ingestion)
2. Control and data **flow**
3. Scheduling
4. Controlling



Data Factory Pipeline (Blob-Extract ---> Trans(Copy) ---> Az SQL[Load])

- Linked Service <- Blob
- **DataSet** <- CSV
- Activity <- Process(CSV 1st Row--> Insert into SQL)
- Linked Service --> SQL Server

Storage Account	Service	This PC
Blob Container	Feature	Drive
Blob - Binary Large Object	Resource	File/Folder

Storage Account

- Blob Container
- Table
- Queue
- File System

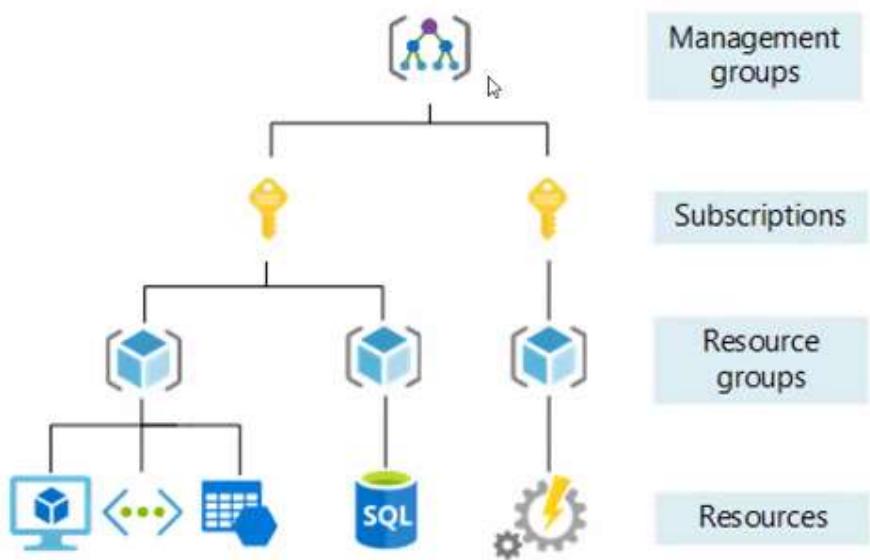
Azure data factory

Storage account

Source

Destination

1. Source(Storage Account --> Container= Input) = Test.csv
2. Sink(Storage Account --> Container = Output)
3. Create ADF Studio & Pipeline
 - a. Launch Studio
 - b. PIPE LINE
 - i. Manage --> Linked Service
 - ii. Author ----> Data Sets(Input & Output)
 - iii. Pipeline --> Activity(Copy) ---> Select Source & Sink
 - iv. Publish
 - v. Debug



Microsoft Azure | Data Factory > df2febyash

Search factory and documentation

Linked services

Linked service defines the connection information to a data store or compute. [Learn more](#)

+ New

Filter by name Annotations : Any

Showing 1 - 2 of 2 items

Name	Type	Related	Annotations
AzureDataLakeStorage1	Azure Data Lake Storage Gen2	0	
AzureSqlDatabase1	Azure SQL Database	0	

Preview experience Off

26°C Smoke 18:59 02-02-2024

Microsoft Azure | Data Factory > df2febyash

Search factory and documentation

Linked services

Linked service defines the connection information to a data store or compute. [Learn more](#)

+ New

Filter by name Annotations : Any

Showing 1 - 2 of 2 items

Name	Type	Related	Annotations
AzureDataLakeStorage1	Azure Data Lake Storage Gen2	0	
AzureSqlDatabase1	Azure SQL Database	0	

Edit linked service

Azure SQL Database [Learn more](#)

Name * AzureSqlDatabase1

Description

Connect via integration runtime * [AutoResolveIntegrationRuntime](#)

Connection string Azure Key Vault

Account selection method

From Azure subscription Enter manually

Fully qualified domain name * serveryash.database.windows.net

Database name *

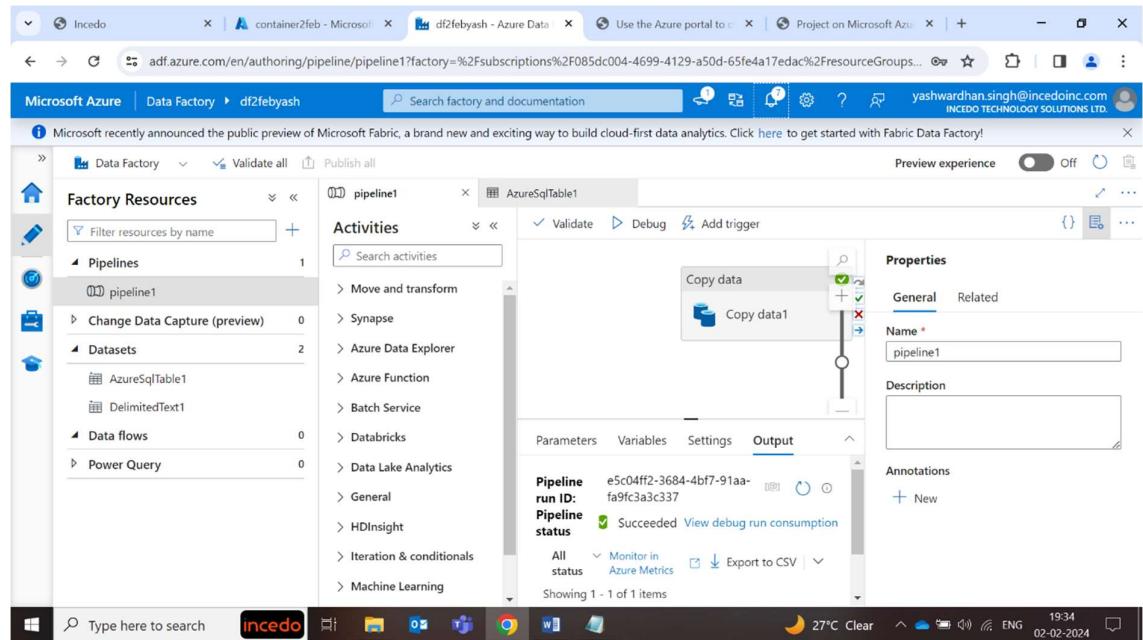
Apply Cancel

Connection successful Test connection

19:03 02-02-2024

The screenshot shows the Microsoft Azure Data Factory pipeline editor. On the left, the 'Factory Resources' sidebar lists 'Pipelines' (1), 'Change Data Capture (preview)' (0), 'Datasets' (2), 'Data flows' (0), and 'Power Query' (0). The 'pipelinel' pipeline is selected. In the main area, a 'Copy data' activity is configured with 'Sink dataset' set to 'AzureSqlTable1'. The 'Properties' pane on the right shows the pipeline name as 'pipeline1'. At the top, there are tabs for 'Activities', 'Validate', 'Validate copy runtime', 'Debug', 'Add trigger', and 'Publish all'. A 'Preview experience' toggle is set to 'Off'. The status bar at the bottom indicates '27°C Clear' and the date '02-02-2024'.

The screenshot shows the 'Publishing' tab in the Microsoft Azure Data Factory interface. It displays a list of pending changes across 'Pipelines', 'Datasets', and 'Linked services'. Under 'Pipelines', 'pipeline1' is listed as '(New)'. Under 'Datasets', 'DelimitedText1' and 'AzureSqlTable1' are listed as '(New)'. Under 'Linked services', 'AzureDataLakeStorage1' is listed as '(Deleted)'. The 'Existing' column shows '-' for all entries. The status bar at the bottom indicates '27°C Clear' and the date '02-02-2024'.



3-2-24

az group create --name Rgyash --location eastus

```
yashwardhan [ ~ ]$ az group create --name Rgyash --location eastus
{
  "id": "/subscriptions/085dc004-4699-4129-a50d-65fe4a17edac/resourceGroups/Rgyash",
  "location": "eastus",
  "managedBy": null,
  "name": "Rgyash",
  "properties": {
    "provisioningState": "Succeeded"
  },
  "tags": null,
  "type": "Microsoft.Resources/resourceGroups"
}
```

```
az network vnet create
  --name Vnet1
  --resource-group Rgyash
  --address-prefixes 10.0.0.0/16
  --subnet-name Subnet1
  --subnet-prefix 10.0.0.0/24
```

```
yashwardhan [ ~ ]$ az network vnet create --name Vnet1 --resource-group Rgyash --address-prefixes 10.0.0.0/16 --subnet-name Subnet1 --subnet-prefix 10.0.0.0/24
{
  "newVNet": {
    "addressSpace": {
      "addressPrefixes": [
        "10.0.0.0/16"
      ]
    },
    "enableDdosProtection": false,
    "etag": "W/\"8671c36e-de52-4b51-8450-9d48e17e24e7\"",
    "id": "/subscriptions/085dc004-4699-4129-a50d-65fe4a17edac/resourceGroups/Rgyash/providers/Microsoft.Network/virtualNetworks/Vnet1",
    "location": "eastus",
    "name": "Vnet1",
    "provisioningState": "Succeeded",
    "resourceGroup": "Rgyash",
    "resourceGuid": "8a8ffee0-7e9e-4df6-aa4c-96a547b9803a",
    "subnets": [
      {
        "addressPrefix": "10.0.0.0/24",
        "delegations": [],
        "etag": "W/\"8671c36e-de52-4b51-8450-9d48e17e24e7\"",
        "id": "/subscriptions/085dc004-4699-4129-a50d-65fe4a17edac/resourceGroups/Rgyash/providers/Microsoft.Network/virtualNetworks/Vnet1/subnets/Subnet1",
        "name": "Subnet1",
        "privateEndpointNetworkPolicies": "Disabled",
        "privateLinkServiceNetworkPolicies": "Enabled",
        "provisioningState": "Succeeded",
        "resourceGroup": "Rgyash",
        "type": "Microsoft.Network/virtualNetworks/subnets"
      }
    ],
    "type": "Microsoft.Network/virtualNetworks",
    "virtualNetworkPeerings": []
  }
}
```

```
az network vnet create \
--name Vnet2 \
--resource-group Rgyash \
--address-prefixes 10.1.0.0/16 \
--subnet-name Subnet1 \
--subnet-prefix 10.1.0.0/24
```

```
yashwardhan [ ~ ]$ az network vnet create \
--name Vnet2 \
--resource-group Rgyash \
--address-prefixes 10.1.0.0/16 \
--subnet-name Subnet1 \
--subnet-prefix 10.1.0.0/24
{
  "newVNet": {
    "addressSpace": {
      "addressPrefixes": [
        "10.1.0.0/16"
      ]
    },
    "enableDdosProtection": false,
    "etag": "W/\"e471c9ea-5437-4c4f-8677-ea7de86000fd\"",
    "id": "/subscriptions/085dc004-4699-4129-a50d-65fe4a17edac/resourceGroups/Rgyash/providers/Microsoft.Network/virtualNetworks/Vnet2",
    "location": "eastus",
    "name": "Vnet2",
    "provisioningState": "Succeeded",
    "resourceGroup": "Rgyash",
    "resourceGuid": "3771702b-4bdd-472f-83dd-b7975f388040",
    "subnets": [
      {
        "addressPrefix": "10.1.0.0/24",
        "delegations": [],
        "etag": "W/\"e471c9ea-5437-4c4f-8677-ea7de86000fd\"",
        "id": "/subscriptions/085dc004-4699-4129-a50d-65fe4a17edac/resourceGroups/Rgyash/providers/Microsoft.Network/virtualNetworks/Vnet2/subnets/Subnet1",
        "name": "Subnet1",
        "privateEndpointNetworkPolicies": "Disabled",
        "privateLinkServiceNetworkPolicies": "Enabled",
        "provisioningState": "Succeeded",
        "resourceGroup": "Rgyash",
        "type": "Microsoft.Network/virtualNetworks/subnets"
      }
    ],
    "type": "Microsoft.Network/virtualNetworks",
    "virtualNetworkPeerings": []
  }
}
```

Get the id for \$Vnet1.

```
vNet1Id=(az network vnet show \
--resource-group Rgyash \
--name Vnet1 \
--query id --out tsv)
```

```
yashwardhan [ ~ ]$ vNet1Id=(az network vnet show \
--resource-group Rgyash \
--name Vnet1 \
--query id --out tsv)
```

```
# Get the id for myVirtualNetwork2.
vNet2Id=(az network vnet show \
--resource-group Rgyash \
--name Vnet2 \
--query id \
--out tsv)
```

```
yashwardhan [ ~ ]$ vNet2Id=(az network vnet show \
--resource-group Rgyash \
--name Vnet2 \
--query id \
--out tsv)
```

```
#VNet1 to Vent2
az network vnet peering create \
--name Vnet1-Vnet2 \
--resource-group Rgyash \
--vnet-name Vnet1 \
--remote-vnet $vNet2Id \
--allow-vnet-access
```

```
yashwardhan [ ~ ]$ az network vnet peering create \
--name Vnet1-Vnet2 \
--resource-group Rgyash \
--vnet-name Vnet1 \
--remote-vnet $vNet2Id \
--allow-vnet-access
{
  "allowForwardedTraffic": false,
  "allowGatewayTransit": false,
  "allowVirtualNetworkAccess": true,
  "doNotVerifyRemoteGateways": false,
  "etag": "W\"c19fc01b-d33f-4d25-b512-e2f0ff33e9a9\"",
  "id": "/subscriptions/085dc004-4699-4129-a50d-65fe4a17edac/resourceGroups/Rgyash/providers/Microsoft.Network/virtualNetworks/Vnet1/virtualNetworkPeerings/Vnet1-Vnet2",
  "name": "Vnet1-Vnet2",
  "peeringState": "Initiated",
  "peeringSyncLevel": "RemoteNotInSync",
  "provisioningState": "Succeeded",
  "remoteAddressSpace": [
    "addressPrefixes": [
      "10.1.0.0/16"
    ]
  ],
  "remoteVirtualNetwork": [
    {
      "id": "/subscriptions/085dc004-4699-4129-a50d-65fe4a17edac/resourceGroups/Rgyash/providers/Microsoft.Network/virtualNetworks/Vnet2",
      "resourceGroup": "Rgyash"
    }
  ],
  "remoteVirtualNetworkAddressSpace": [
    "addressPrefixes": [
      "10.1.0.0/16"
    ]
  ],
  "resourceGroup": "Rgyash",
  "resourceGuid": "bdfef8ecb-3543-0ad9-2991-21321881007a",
  "type": "Microsoft.Network/virtualNetworks/virtualNetworkPeerings",
  "useRemoteGateways": false
}
```

```
az network vnet peering create \
--name Vnet2-Vnet1 \
--resource-group Rgyash \
--vnet-name Vnet2 \
--remote-vnet $vNet1Id \
--allow-vnet-access
```

```
yashwardhan [ ~ ]$ az network vnet peering create \
--name Vnet2-Vnet1 \
--resource-group Rgyash \
--vnet-name Vnet2 \
--remote-vnet $vNet1Id \
--allow-vnet-access
{
  "allowForwardedTraffic": false,
  "allowGatewayTransit": false,
  "allowVirtualNetworkAccess": true,
  "doNotVerifyRemoteGateways": false,
  "etag": "W\"a29081f1-33f0-481a-a030-545c1950a26f\"",
  "id": "/subscriptions/085dc004-4699-4129-a50d-65fe4a17edac/resourceGroups/Rgyash/providers/Microsoft.Network/virtualNetworks/Vnet2/virtualNetworkPeerings/Vnet2-Vnet1",
  "name": "Vnet2-Vnet1",
  "peeringState": "Connected",
  "peeringSyncLevel": "FullyInSync",
  "provisioningState": "Succeeded",
  "remoteAddressSpace": {
    "addressPrefixes": [
      "10.0.0.0/16"
    ]
  },
  "remoteVirtualNetwork": {
    "id": "/subscriptions/085dc004-4699-4129-a50d-65fe4a17edac/resourceGroups/Rgyash/providers/Microsoft.Network/virtualNetworks/Vnet1",
    "resourceGroup": "Rgyash"
  },
  "remoteVirtualNetworkAddressSpace": {
    "addressPrefixes": [
      "10.0.0.0/16"
    ]
  },
  "resourceGroup": "Rgyash",
  "resourceGuid": "bdfe8ecb-3543-0ad9-2991-21321881007a",
  "type": "Microsoft.Network/virtualNetworks/virtualNetworkPeerings",
  "useRemoteGateways": false
}
```

```
az network vnet peering show \
--name Vnet1-Vnet2 \
--resource-group Rgyash \
--vnet-name Vnet1 \
--query peeringState
```

```
yashwardhan [ ~ ]$ az network vnet peering show \
--name Vnet1-Vnet2 \
--resource-group Rgyash \
--vnet-name Vnet1 \
--query peeringState
"Connected"
```

```
az vm create \
--resource-group Rgyash \
--name myVm1 \
--image Ubuntu2204 \
--public-ip-sku Standard \
--vnet-name Vnet1 \
--subnet Subnet1 \
```

```
--ssh-key-value ~/.ssh/mujahed_lock.pub \
--no-wait
```

```
yashwardhan [ ~ ]$ ssh-keygen -t rsa -b 2048 -f ~/.ssh/yash_lock
Generating public/private rsa key pair.
Created directory '/home/yashwardhan/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/yashwardhan/.ssh/yash_lock
Your public key has been saved in /home/yashwardhan/.ssh/yash_lock.pub
The key fingerprint is:
SHA256:Qxu+u80+aIhEKanIsSOCEdZlGsV3t+a/oiwrh5uQ4 yashwardhan@SandboxHost-638425588755369208
The key's randomart image is:
+--- [RSA 2048] ---+
| .+o+... |
| o.*+ ... |
| Bo...o ..o. |
| +o ..o...o. |
| ... So |
| . o.. . |
| .E+ o... . |
| oo* ++o o |
| o+o+o+=*. |
+--- [SHA256] ---+
```

```
az vm create \
--resource-group Rgyash \
--name myVm2 \
--image Ubuntu2204 \
--public-ip-sku Standard \
--vnet-name Vnet2 \
--subnet Subnet1 \
--ssh-key-value ~/.ssh/yash_lock.pub
```

```
yashwardhan [ ~ ]$ az vm create \
--resource-group Rgyash \
--name myVm2 \
--image Ubuntu2204 \
--public-ip-sku Standard \
--vnet-name Vnet2 \
--subnet Subnet1 \
--ssh-key-value ~/.ssh/yash_lock.pub --generate-ssh-keys

{
  "fqdns": "",
  "id": "/subscriptions/085dc004-4699-4129-a50d-65fe4a17edac/resourceGroups/Rgyash/providers/Microsoft.Compute/virtualMachines/myVm2",
  "location": "eastus",
  "macAddress": "60-45-BD-EB-E0-3C",
  "powerState": "VM running",
  "privateIpAddress": "10.1.0.4",
  "publicIpAddress": "20.127.0.229",
  "resourceGroup": "Rgyash",
  "zones": ""
```

```
ssh -i ~/.ssh/yash_lock yash@20.127.0.229 (ip of vm2)
```

```
yashwardhan [ ~ ]$ ssh -i ~/.ssh/yash lock yash@20.127.0.229
The authenticity of host '20.127.0.229 (20.127.0.229)' can't be established.
ED25519 key fingerprint is SHA256:PIX6U78iMvk3tIl3QcVsAwKdnNuOGRKZfyOAZBoyKp4.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])?
Host key verification failed.
```

```
#Ping to VM1 ip  
ping 10.0.0.4 -c 4
```

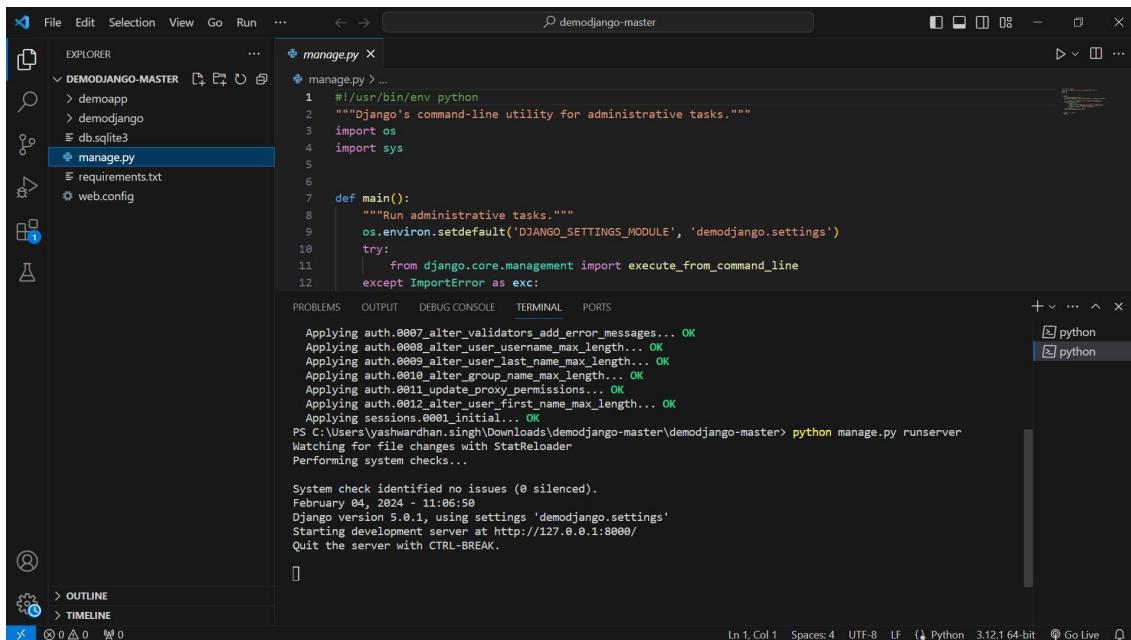
```
yashwardhan [ ~ ]$ ping 20.121.38.14  
PING 20.121.38.14 (20.121.38.14) 56(84) bytes of data.
```

```
az group delete --name $Rg --yes
```

04/02/2024

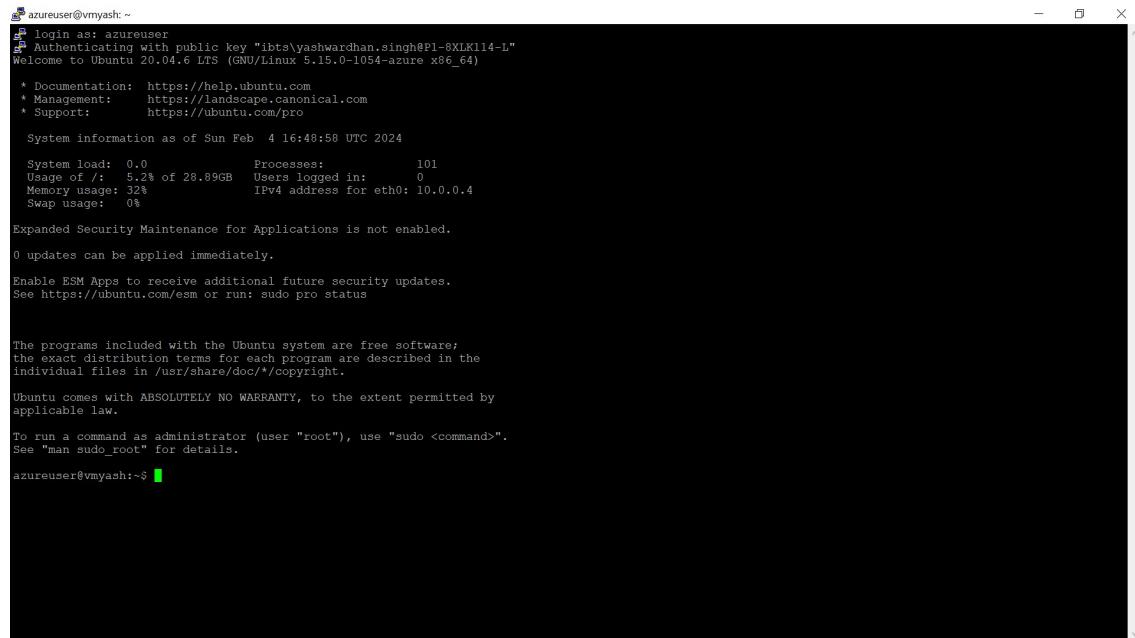
Project Title:

"Efficient Django Deployment: Dockerized Application on Azure VM"





```
yashwardhan [ ~ ]$ ssh-keygen -t rsa -b 2048 -f ~/.ssh/my-ssh-key
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/yashwardhan/.ssh/my-ssh-key
Your public key has been saved in /home/yashwardhan/.ssh/my-ssh-key.pub
The key fingerprint is:
SHA256:jIMGHSkry19ml/AhXoojVX70fRmS+HUF4t7a1XefWeM yashwardhan@SandboxHost-638426230841341143
The key's randomart image is:
+---[RSA 2048]----+
|   oo . .o...o|
| .oo+.o...o..o. |
| oo+.*.o...oo. |
|oo.+ B.* o.. . |
|o.o B + S    . .*|
|   o .     o..O|
|       . .E.|
|           |
|           |
+---[SHA256]-----+
```



```

Preparing to unpack .../1-containerd.io_1.6.28-1_amd64.deb ...
Unpacking containerd.io (1.6.28-1) ...
Selecting previously unselected package docker-buildx-plugin.
Preparing to unpack .../2-docker-buildx-plugin_0.12.1-1~ubuntu.20.04-focal_amd64
deb ...
Unpacking docker-buildx-plugin (0.12.1-1~ubuntu.20.04-focal) ...
Selecting previously unselected package docker-ce-cli.
Preparing to unpack .../3-docker-ce-cli_5%3a25.0.2-1~ubuntu.20.04-focal_amd64.de
b ...
Unpacking docker-ce-cli (5:25.0.2-1~ubuntu.20.04-focal) ...
Selecting previously unselected package docker-ce.
Preparing to unpack .../4-docker-ce_5%3a25.0.2-1~ubuntu.20.04-focal_amd64.deb ...
Unpacking docker-ce (5:25.0.2-1~ubuntu.20.04-focal) ...
Selecting previously unselected package docker-ce-rootless-extras.
Preparing to unpack .../5-docker-ce-rootless-extras_5%3a25.0.2-1~ubuntu.20.04-fo
cal_amd64.deb ...
Unpacking docker-ce-rootless-extras (5:25.0.2-1~ubuntu.20.04-focal) ...
Selecting previously unselected package docker-compose-plugin.
Preparing to unpack .../6-docker-compose-plugin_2.24.5-1~ubuntu.20.04-focal_amd6
4.deb ...
Unpacking docker-compose-plugin (2.24.5-1~ubuntu.20.04-focal) ...
Selecting previously unselected package slirp4netns.
Preparing to unpack .../7-slirp4netns_0.4.3-1_amd64.deb ...
Unpacking slirp4netns (0.4.3-1) ...
Setting up slirp4netns (0.4.3-1) ...
Setting up docker-buildx-plugin (0.12.1-1~ubuntu.20.04-focal) ...
Setting up containerd.io (1.6.28-1) ...
Setting up containerd.io.service - /lib/systemd/system/containerd.service →
/lib/systemd/system/multi-user.target.wants/containerd.service
Setting up docker-compose-plugin (2.24.5-1~ubuntu.20.04-focal) ...
Setting up docker-ce-cli (5:25.0.2-1~ubuntu.20.04-focal) ...
Setting up pigz (2.4-1) ...
Setting up docker-ce-rootless-extras (5:25.0.2-1~ubuntu.20.04-focal) ...
Setting up docker-ce (5:25.0.2-1~ubuntu.20.04-focal) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /i
b/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /lib/sy
stemd/system/docker.socket.
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for systemd (245.4-4ubuntu3.23) ...
azureuser@vmyashi:~$ sudo systemctl start docker
azureuser@vmyashi:~$ sudo systemctl enable docker
Synchronizing state of docker.service with SysV service script with /lib/systemd
/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable docker
azureuser@vmyashi:~$ docker --version
Docker version 25.0.2, build 29cf629
azureuser@vmyashi:~$ █

```

Module 2: Perform T-SQL Operations on Restored Database (50 Marks)

1. Retrieve a list of customers along with their total order amounts.

```

SELECT
    c.CustomerID,
    c.FirstName,
    c.LastName,
    SUM(o.TotalDue) AS TotalOrderAmount
FROM
    SalesLT.Customer c
JOIN
    SalesLT.SalesOrderHeader o ON c.CustomerID = o.CustomerID
GROUP BY
    c.CustomerID, c.FirstName, c.LastName
ORDER BY
    c.CustomerID;

```

```

6   FKUMI
7   SalesLT.Customer c
8   JOIN
9   SalesLT.SalesOrderHeader o ON c.CustomerID = o.CustomerID
10  GROUP BY
11  c.CustomerID, c.FirstName, c.LastName
12  ORDER BY
13  c.CustomerID;

```

CustomerID	FirstName	LastName	TotalOrderAmount
29485	Catherine	Abel	43962.7901
29531	Cory	Booth	7330.8972

2. Display product information along with the number of units sold for each product.

```

SELECT
p.ProductID,
p.Name AS ProductName,
p.ProductNumber,
p.Color,
SUM(od.OrderQty) AS TotalUnitsSold
FROM
SalesLT.Product p
JOIN
SalesLT.SalesOrderDetail od ON p.ProductID = od.ProductID
JOIN
SalesLT.SalesOrderHeader oh ON od.SalesOrderID = oh.SalesOrderID
GROUP BY
p.ProductID, p.Name, p.ProductNumber, p.Color
ORDER BY
p.ProductID;

```

```

    JOIN
10  SalesLT.SalesOrderDetail od ON p.ProductID = od.ProductID
11  JOIN
12  SalesLT.SalesOrderHeader oh ON od.SalesOrderID = oh.SalesOrderID
13  GROUP BY
14  p.ProductID, p.Name, p.ProductNumber, p.Color
15  ORDER BY
16  p.ProductID;

```

ProductID	ProductName	ProductNumber	Color	TotalUnits
707	Sport-100 Helmet, R...	HL-U509-R	Red	35
708	Sport-100 Helmet, Bl...	HL-U509	Black	51

3. Find employees who have the same manager.

Data not available.

4. List all customers who have never placed an order.

```

SELECT
c.CustomerID,
c.FirstName,
c.LastName
FROM
SalesLT.Customer c
LEFT JOIN
SalesLT.SalesOrderHeader o ON c.CustomerID = o.CustomerID
WHERE
o.CustomerID IS NULL;

```

The screenshot shows the Microsoft Azure portal interface for a SQL database named 'dbyash'. The top navigation bar includes 'Microsoft Azure', a search bar, and user information 'yashwardhan.singh@inc... INCEDO TECHNOLOGY SOLUTIONS'. Below the navigation is a breadcrumb trail: Home > Microsoft.SQLDatabase.newDatabaseExistingServer_4d93904160ff4b45 | Overview > dbyash (snyash/dbyash). The main area is titled 'dbyash (snyash/dbyash) | Query editor (preview)' and shows a query editor window. The query editor has tabs for 'Query 1' (selected), 'Run', 'Cancel query', 'Save query', 'Export data as', and 'Show only Editor'. The query itself is a SELECT statement:

```

1  SELECT
2    c.CustomerID,
3    c.FirstName,
4    c.LastName
5  FROM
6    SalesLT.Customer c
7  LEFT JOIN
8    SalesLT.SalesOrderHeader o ON c.CustomerID = o.CustomerID
9  WHERE

```

The results tab displays a table with four rows of customer data:

CustomerID	FirstName	LastName
1	Orlando	Gee
2	Keith	Harris
3	Donna	Carreras
4	Janet	Gates

A message at the bottom of the results pane says 'Query succeeded | 1s'.

5. Retrieve the total sales amount for each product category.

```

SELECT
pc.ProductCategoryID,
pc.Name AS CategoryName,
SUM(od.OrderQty * od.UnitPrice) AS TotalSalesAmount
FROM
SalesLT.ProductCategory pc
JOIN
SalesLT.Product p ON pc.ProductCategoryID = p.ProductCategoryID
JOIN
SalesLT.SalesOrderDetail od ON p.ProductID = od.ProductID
GROUP BY
pc.ProductCategoryID, pc.Name
ORDER BY
pc.ProductCategoryID;

```

The screenshot shows the Azure Data Studio interface. On the left, there's a sidebar titled 'dbyash (dbadmin)' which lists 'Tables', 'Views', and 'Stored Procedures'. A message box says 'Showing limited object explorer here. For full capability please click here to open Azure Data Studio.' In the main area, a query window titled 'Query 1' contains the following SQL code:

```

1  SELECT
2    pc.ProductCategoryID,
3    pc.Name AS CategoryName,
4    SUM(od.OrderQty * od.UnitPrice) AS TotalSalesAmount
5  FROM
6    SalesLT.ProductCategory pc
7  JOIN
8    SalesLT.Product p ON pc.ProductCategoryID = p.ProductCategoryID

```

The results tab shows a table with three columns: 'ProductCategoryID', 'CategoryName', and 'TotalSalesAmount'. The data is as follows:

ProductCategoryID	CategoryName	TotalSalesAmount
5	Mountain Bikes	173085.8460
6	Road Bikes	185513.0436

6. Display the names of employees and their direct managers. Data not available

7. Show the order details with product names for a specific customer.

```

SELECT
oh.SalesOrderID,
od.ProductID,
p.Name AS ProductName,
od.OrderQty,
od.UnitPrice,
od.LineTotal
FROM
SalesLT.Customer c
JOIN
SalesLT.SalesOrderHeader oh ON c.CustomerID = oh.CustomerID
JOIN
SalesLT.SalesOrderDetail od ON oh.SalesOrderID = od.SalesOrderID
JOIN
SalesLT.Product p ON od.ProductID = p.ProductID
WHERE
c.CustomerID = 29485;

```

Showing limited object explorer here. For full capability please click here to open Azure Data Studio.

```

1 SELECT
2     oh.SalesOrderID,
3     od.ProductID,
4     p.Name AS ProductName,
5     od.OrderQty,
6     od.UnitPrice,
7     od.LineTotal
8 END

```

SalesOrderID	ProductID	ProductName	OrderQty	UnitPrice	LineTotal
71782	714	Long-Sleeve Logo Jer...	3	29.9940	89.982000
71782	956	Touring-1000 Yellow, ...	3	1430.4420	4291.326000

8. List customers who have made purchases in the last 30 days

```

SELECT DISTINCT
c.CustomerID,
c.FirstName,
c.LastName
FROM
SalesLT.Customer c
JOIN
SalesLT.SalesOrderHeader oh ON c.CustomerID = oh.CustomerID
WHERE
oh.OrderDate >= DATEADD(day, -30, GETDATE());

```

Showing limited object explorer here. For full capability please click here to open Azure Data Studio.

```

1 SELECT DISTINCT
2     c.CustomerID,
3     c.FirstName,
4     c.LastName
5     FROM
6     SalesLT.Customer c
7     JOIN

```

Query succeeded: Affected rows: 0

9. Find employees who do not have any direct reports.

Data Not available

10. Retrieve all products along with their average selling prices

```
SELECT
    p.ProductID,
    p.Name AS ProductName,
    AVG(od.UnitPrice) AS AverageSellingPrice
FROM
    SalesLT.Product p
JOIN
    SalesLT.SalesOrderDetail od ON p.ProductID = od.ProductID
GROUP BY
    p.ProductID, p.Name
ORDER BY
    p.ProductID;
```

The screenshot shows the Azure Data Studio interface. The top navigation bar includes 'Home', 'Microsoft.SQLDatabase.newDatabaseExistingServer_4d93904160ff4b45 | Overview', 'dbyash (snyash/dbyash)', and a user icon. Below the navigation is a toolbar with 'Login', 'New Query', 'Open query', 'Feedback', and 'Getting started'. The main area is titled 'Query 1' and contains the SQL code from step 10. The code is highlighted with syntax coloring. Below the code is a results pane titled 'Results' which displays a table with two rows of data:

ProductID	ProductName	AverageSellingPrice
707	Sport-100 Helmet, Red	20.9940
708	Sport-100 Helmet, Black	20.6441

11. Find the order with the highest total amount.

```
SELECT TOP 1
    oh.SalesOrderID,
    oh.OrderDate,
    SUM(od.LineTotal) AS TotalAmount
FROM
    SalesLT.SalesOrderHeader oh
JOIN
    SalesLT.SalesOrderDetail od ON oh.SalesOrderID = od.SalesOrderID
```

```

GROUP BY
oh.SalesOrderID, oh.OrderDate
ORDER BY
TotalAmount DESC;

```

The screenshot shows the Azure Data Studio interface. On the left, there's a sidebar titled 'dbyash (dbadmin)' with options for Tables, Views, and Stored Procedures. The main area is titled 'Query 1' and contains the following SQL code:

```

1  SELECT TOP 1
2  oh.SalesOrderID,
3  oh.OrderDate,
4  SUM(od.LineTotal) AS TotalAmount
5  FROM
6  SalesLT.SalesOrderHeader oh
7  JOIN

```

Below the code, the 'Results' tab is selected, showing a single row of data:

SalesOrderID	OrderDate	TotalAmount
71784	2008-06-01T00:00:00.0000000	89869.276314

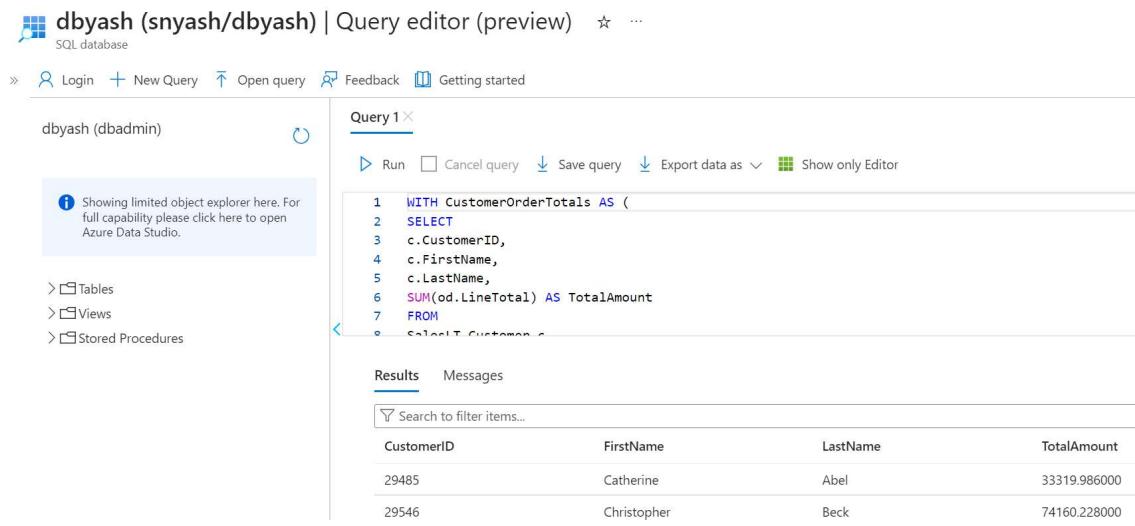
12. Display customers who have placed orders with a total amount greater than the average.

```

WITH CustomerOrderTotals AS (
SELECT
c.CustomerID,
c.FirstName,
c.LastName,
SUM(od.LineTotal) AS TotalAmount
FROM
SalesLT.Customer c
JOIN
SalesLT.SalesOrderHeader oh ON c.CustomerID = oh.CustomerID
JOIN
SalesLT.SalesOrderDetail od ON oh.SalesOrderID = od.SalesOrderID
GROUP BY
c.CustomerID, c.FirstName, c.LastName
)
SELECT
CustomerID,
FirstName,
LastName,
TotalAmount
FROM
CustomerOrderTotals
WHERE

```

```
TotalAmount > (SELECT AVG(TotalAmount) FROM CustomerOrderTotals);
```



The screenshot shows the Azure Data Studio interface. The top navigation bar includes 'Login', 'New Query', 'Open query', 'Feedback', and 'Getting started'. The left sidebar shows a user 'dbyash (dbadmin)' and a message: 'Showing limited object explorer here. For full capability please click here to open Azure Data Studio.' Below this are sections for 'Tables', 'Views', and 'Stored Procedures'. The main area is titled 'Query 1' and contains the following SQL code:

```
1 WITH CustomerOrderTotals AS (
2     SELECT
3         c.CustomerID,
4         c.FirstName,
5         c.LastName,
6         SUM(od.LineTotal) AS TotalAmount
7     FROM
8         SalesLT.Customer c
9         JOIN SalesLT.OrderDetail od ON c.CustomerID = od.CustomerID
10    GROUP BY c.CustomerID, c.FirstName, c.LastName)
```

The results pane shows a table with four columns: CustomerID, FirstName, LastName, and TotalAmount. There are two rows of data:

CustomerID	FirstName	LastName	TotalAmount
29485	Catherine	Abel	33319.986000
29546	Christopher	Beck	74160.228000

13. List products with prices higher than the average product price.

```
WITH ProductPrices AS (
    SELECT
        ProductID,
        Name AS ProductName,
        ListPrice
    FROM
        SalesLT.Product
)
SELECT
    ProductID,
    ProductName,
    ListPrice
FROM
    ProductPrices
WHERE
    ListPrice > (SELECT AVG(ListPrice) FROM ProductPrices);
```

The screenshot shows the Azure Data Studio interface. On the left, there's a sidebar with a connection named 'dbyash (snyash/dbyash)' and a note about limited object explorer. Below it are 'Tables', 'Views', and 'Stored Procedures'. The main area is titled 'Query 1' and contains the following T-SQL code:

```

1 WITH ProductPrices AS (
2     SELECT
3         ProductID,
4         Name AS ProductName,
5         ListPrice
6     FROM
7         SalesLT.Product
8     )

```

Below the code, there are 'Results' and 'Messages' tabs, and a search bar. The results table shows two rows of data:

ProductID	ProductName	ListPrice
680	HL Road Frame - Black, 58	1431.5000
706	HL Road Frame - Red, 58	1431.5000

14. Retrieve orders placed by employees who have a specific job title.

Data not available

15. Display customers who have placed orders for a specific product category

```

SELECT
    c.CustomerID,
    c.FirstName,
    c.LastName
FROM
    SalesLT.Customer c
JOIN
    SalesLT.SalesOrderHeader oh ON c.CustomerID = oh.CustomerID
JOIN
    SalesLT.SalesOrderDetail od ON oh.SalesOrderID = od.SalesOrderID
JOIN
    SalesLT.Product p ON od.ProductID = p.ProductID
JOIN
    SalesLT.ProductCategory pc ON p.ProductCategoryID =
        pc.ProductCategoryID
WHERE
    pc.ProductCategoryID = 22;

```

The screenshot shows the Azure Data Studio interface. At the top, it says "dbyash (snyash/dbyash) | Query editor (preview)" with a star icon and three dots. Below the header are navigation links: "Login", "New Query", "Open query", "Feedback", and "Getting started". The main area has a title "Query 1" with a close button. Below the title are toolbar buttons for "Run", "Cancel query", "Save query", "Export data as", and "Show only Editor". The query editor contains the following SQL code:

```
1 SELECT
2     c.CustomerID,
3     c.FirstName,
4     c.LastName
5 FROM
6     SalesLT.Customer c
7 JOIN
```

Below the code, there are two tabs: "Results" and "Messages", with "Messages" being the active tab. A message states "Query succeeded: Affected rows: 0".

16. Find employees with salaries greater than the average salary in their department.

Data Not available

17. List customers who have placed orders before a specific date. Enough data not available for query

```
SELECT DISTINCT
c.CustomerID,
c.FirstName,
c.LastName
FROM
SalesLT.Customer c
JOIN
SalesLT.SalesOrderHeader oh ON c.CustomerID = oh.CustomerID
WHERE
oh.OrderDate < 2008-07-02;
```

The screenshot shows the Azure Data Studio interface. On the left, there's a sidebar for the database 'dbyash (snyash/dbyash) | SQL database'. It has sections for 'Tables', 'Views', and 'Stored Procedures'. A message box says: 'Showing limited object explorer here. For full capability please click here to open Azure Data Studio.' At the top, there are navigation links: 'Login', 'New Query', 'Open query', 'Feedback', and 'Getting started'. Below the sidebar is the main query editor area. The title bar says 'Query 1'. The toolbar includes 'Run', 'Cancel query', 'Save query', 'Export data as', and 'Show only Editor'. The query itself is:

```
1 SELECT DISTINCT
2     c.CustomerID,
3     c.FirstName,
4     c.LastName
5 FROM
6     SalesLT.Customer c
7 JOIN
```

Below the query, there are tabs for 'Results' and 'Messages'. The message tab shows: 'Query succeeded: Affected rows: 0'

18. Retrieve the order with the highest quantity of a specific product

```
SELECT TOP 1
od.SalesOrderID,
od.ProductID,
p.Name AS ProductName,
SUM(od.OrderQty) AS TotalQuantity
FROM
SalesLT.SalesOrderDetail od
JOIN
SalesLT.Product p ON od.ProductID = p.ProductID
WHERE
od.ProductID = 714
GROUP BY
od.SalesOrderID, od.ProductID, p.Name
ORDER BY
TotalQuantity DESC;
```

The screenshot shows the Azure Data Studio interface. On the left, there's a sidebar with a 'Tables' section containing 'Tables', 'Views', and 'Stored Procedures'. A message box says: 'Showing limited object explorer here. For full capability please click here to open Azure Data Studio.' At the top, there are navigation links: 'Login', 'New Query', 'Open query', 'Feedback', and 'Getting started'. Below the navigation is a toolbar with 'Run', 'Cancel query', 'Save query', 'Export data as', and 'Show only Editor'. The main area is titled 'Query 1' and contains the following T-SQL code:

```

1  SELECT TOP 1
2    od.SalesOrderID,
3    od.ProductID,
4    p.Name AS ProductName,
5    SUM(od.OrderQty) AS TotalQuantity
6  FROM
7    SalesLT.SalesOrderDetail od

```

Below the code, there are tabs for 'Results' and 'Messages', with 'Results' selected. The results table has columns: SalesOrderID, ProductID, ProductName, and TotalQuantity. One row is shown:

SalesOrderID	ProductID	ProductName	TotalQuantity
71784	714	Long-Sleeve Logo Jersey, M	9

19. Display products with prices lower than the lowest product price in a specific category.

```

WITH ProductPrices AS (
SELECT
p.ProductID,
p.Name AS ProductName,
p.ListPrice,
pc.ProductCategoryID
FROM
SalesLT.Product p
JOIN
SalesLT.ProductCategory pc ON p.ProductCategoryID =
pc.ProductCategoryID
WHERE
pc.ProductCategoryID = 11
)
SELECT
ProductID,
ProductName,
ListPrice
FROM
ProductPrices
WHERE
ListPrice < (SELECT MIN(ListPrice) FROM ProductPrices);

```

The screenshot shows the Azure Data Studio interface. On the left, there's a sidebar titled 'dbyash (dbadmin)' with options for Tables, Views, and Stored Procedures. The main area is titled 'Query 1' and contains the following SQL code:

```
1 WITH ProductPrices AS (
2     SELECT
3         p.ProductID,
4         p.Name AS ProductName,
5         p.ListPrice,
6         pc.ProductCategoryID
7     FROM
```

Below the code, there are 'Results' and 'Messages' tabs, and a status message: 'Query succeeded: Affected rows: 0'.

20. Find employees who have the same job title as their manager.

Data not available

21. Combine results from two queries to get a list of unique customer and employee names.

Data Not Available

22. Retrieve product names that are common in two different product categories.

Not enough data available

```
SELECT
    p.Name AS ProductName
FROM
    SalesLT.Product p
JOIN
    SalesLT.ProductCategory pc1 ON p.ProductCategoryID =
        pc1.ProductCategoryID
JOIN
    SalesLT.ProductCategory pc2 ON p.ProductCategoryID =
        pc2.ProductCategoryID
WHERE
    pc1.ProductCategoryID <> pc2.ProductCategoryID;
```

23. Display the names of employees and customers in a single result set.

Data Not Available

24.List products that are in stock or have been discontinued.

Data Not Available

25.Combine the results of two queries to find unique products ordered by a specific customer.

```
SELECT DISTINCT
    p.ProductID,
    p.Name AS ProductName
FROM
    SalesLT.Product p
JOIN
    SalesLT.SalesOrderDetail od ON p.ProductID = od.ProductID
JOIN
    SalesLT.SalesOrderHeader oh ON od.SalesOrderID = oh.SalesOrderID
WHERE
    oh.CustomerID = 29485
UNION
SELECT DISTINCT
    p.ProductID,
    p.Name AS ProductName
FROM
    SalesLT.Product p
JOIN
    SalesLT.SalesOrderDetail od ON p.ProductID = od.ProductID
ORDER BY
    ProductID;
```

The screenshot shows the Azure Data Studio interface with the following details:

- Header:** dbyash (snyash/dbyash) | Query editor (preview) ☆ ...
- Top Bar:** Login, New Query, Open query, Feedback, Getting started
- Left Sidebar:** Shows a user profile icon and the database name "dbyash (dbadmin)". Below it, there's a note about limited object explorer and a link to open Azure Data Studio. Under "Tables", "Views", and "Stored Procedures", there are expandable arrows.
- Query Editor:** A tab labeled "Query 1" is active. It contains the SQL code from the previous section. Below the code are buttons for Run, Cancel query, Save query, Export data as, and Show only Editor.
- Results Tab:** Labeled "Results". It shows a table with two rows of data:

ProductID	ProductName
707	Sport-100 Helmet, Red
708	Sport-100 Helmet, Black
- Messages Tab:** Labeled "Messages". It is currently empty.

26.Retrieve orders placed by customers and employees in a single result set.

Data not available

27.Display products that are either in a specific category or have a specific safety stock level.

Data not Available

28.List customers who have placed orders and employees who have direct reports in a single result set.

Data Not Available

29.Retrieve products that are in stock in one location and out of stock in another.

Data Not Available

30.Combine information about employees who are managers and employees who have managers

Data Not Available INTERMEDIATE

31.Retrieve a list of customers along with the names of the products they have purchased.

```
SELECT
    c.CustomerID,
    c.FirstName,
    c.LastName,
    p.Name AS ProductName
FROM
    SalesLT.Customer c
JOIN
    SalesLT.SalesOrderHeader oh ON c.CustomerID = oh.CustomerID
JOIN
    SalesLT.SalesOrderDetail od ON oh.SalesOrderID = od.SalesOrderID
JOIN
    SalesLT.Product p ON od.ProductID = p.ProductID
ORDER BY
    c.CustomerID, p.ProductID;
```

The screenshot shows the Azure Data Studio interface. On the left, there's a sidebar titled 'dbyash (dbadmin)' which includes a note about limited object explorer and links to 'Tables', 'Views', and 'Stored Procedures'. The main area is titled 'Query 1' and contains a SQL script:

```

1 SELECT
2     c.CustomerID,
3     c.FirstName,
4     c.LastName,
5     p.Name AS ProductName
6 FROM
7     SalesLT.Customer c
8     JOIN SalesLT.Product p ON c.CustomerID = p.ProductID
9 
```

Below the script, there are tabs for 'Results' and 'Messages', and a search bar. The results table shows two rows of data:

CustomerID	FirstName	LastName	ProductName
29485	Catherine	Abel	Sport-100 Helmet, Red
29485	Catherine	Abel	Sport-100 Helmet, Black

32. Display employees who have the same manager, including indirect reports.

Data Not Available

33. Find orders with multiple products and display the product names.

```

SELECT
    oh.SalesOrderID,
    COUNT(DISTINCT od.ProductID) AS NumberOfProducts,
    STRING_AGG(p.Name, ', ') AS ProductNames
FROM
    SalesLT.SalesOrderHeader oh
JOIN
    SalesLT.SalesOrderDetail od ON oh.SalesOrderID = od.SalesOrderID
GROUP BY
    oh.SalesOrderID
HAVING
    COUNT(DISTINCT od.ProductID) > 1;

```

The screenshot shows the Azure Data Studio interface with a query editor titled 'Query 1'. The query is as follows:

```

1 SELECT
2     oh.SalesOrderID,
3     COUNT(DISTINCT od.ProductID) AS NumberOfProducts,
4     STRING_AGG(p.Name, ', ') AS ProductNames
5 FROM
6     SalesLT.SalesOrderHeader oh
7 JOIN
8     SalesLT.SalesOrderDetail od ON oh.SalesOrderID = od.SalesOrderID

```

The results table has three columns: SalesOrderID, NumberOfProducts, and ProductNames. The data is:

SalesOrderID	NumberOfProducts	ProductNames
71774	2	ML Road Frame-W - Yellow, 48, ML Road Fram
71780	29	LL Mountain Frame - Silver, 44, LL Mountain Fr

34.List customers along with the names of the salespeople who handled their orders.

Data Not Available

35.Retrieve a list of products along with the names of suppliers.

Data Not Available

36.Display customers who have placed orders and the products they have purchased, including product details.

```

SELECT
c.CustomerID,
c.FirstName,
c.LastName,
oh.SalesOrderID,
p.ProductID,
p.Name AS ProductName,
od.OrderQty,
od.UnitPrice
FROM
SalesLT.Customer c
JOIN
SalesLT.SalesOrderHeader oh ON c.CustomerID = oh.CustomerID
JOIN
SalesLT.SalesOrderDetail od ON oh.SalesOrderID = od.SalesOrderID
JOIN
SalesLT.Product p ON od.ProductID = p.ProductID
ORDER BY
c.CustomerID, oh.SalesOrderID, p.ProductID;

```

The screenshot shows the Azure Data Studio interface. On the left, there's a sidebar titled 'dbyash (dbadmin)' with options like 'Tables', 'Views', and 'Stored Procedures'. The main area is titled 'Query 1' and contains the following SQL code:

```

1  SELECT
2    c.CustomerID,
3    c.FirstName,
4    c.LastName,
5    oh.SalesOrderID,
6    p.ProductID,
7    p.Name AS ProductName,
8    od.OrderQty

```

Below the code, there are 'Results' and 'Messages' tabs. The 'Results' tab shows a table with the following data:

CustomerID	FirstName	LastName	SalesOrderID	ProductID	ProductName
29485	Catherine	Abel	71782	707	Sport-100 Helmet, R...
29485	Catherine	Abel	71782	708	Sport-100 Helmet, Bl...

37. Find orders where multiple employees were involved, showing the employee names.

Data Not Available

38. List products that have similar names but belong to different categories.

Data Not Enough

39. Retrieve a list of employees along with their training courses and training dates.

Data Not Available

40. Display customers who have placed orders and the total quantity of each product ordered.

```

SELECT
c.CustomerID,
c.FirstName,
c.LastName,
p.ProductID,
p.Name AS ProductName,
SUM(od.OrderQty) AS TotalQuantity
FROM
SalesLT.Customer c
JOIN
SalesLT.SalesOrderHeader oh ON c.CustomerID = oh.CustomerID
JOIN
SalesLT.SalesOrderDetail od ON oh.SalesOrderID = od.SalesOrderID
JOIN
SalesLT.Product p ON od.ProductID = p.ProductID

```

```

GROUP BY
c.CustomerID, c.FirstName, c.LastName, p.ProductID, p.Name
ORDER BY
c.CustomerID, p.ProductID;

```

dbyash (snyash/dbyash) | Query editor (preview) ☆ ...

» [Login](#) [New Query](#) [Open query](#) [Feedback](#) [Getting started](#)

dbyash (dbadmin)

Showing limited object explorer here. For full capability please click here to open Azure Data Studio.

> [Tables](#)
> [Views](#)
> [Stored Procedures](#)

Query 1 ×

Run Cancel query Save query Export data as Show only Editor

```

1  SELECT
2  c.CustomerID,
3  c.FirstName,
4  c.LastName,
5  p.ProductID,
6  p.Name AS ProductName,
7  SUM(od.OrderQty) AS TotalQuantity
8  FROM

```

Results Messages

Search to filter items...

CustomerID	FirstName	LastName	ProductID	ProductName	TotalQuantity
29485	Catherine	Abel	707	Sport-100 Helmet, Red	3
29485	Catherine	Abel	708	Sport-100 Helmet, Bla...	7

41. Find customers who have made more purchases than the average number of purchases.

Not Enough Data

```

WITH CustomerPurchaseCounts AS (
SELECT
c.CustomerID,
COUNT(DISTINCT oh.SalesOrderID) AS PurchaseCount
FROM
SalesLT.Customer c
JOIN
SalesLT.SalesOrderHeader oh ON c.CustomerID = oh.CustomerID
JOIN
SalesLT.SalesOrderDetail od ON oh.SalesOrderID = od.SalesOrderID
GROUP BY
c.CustomerID
)
SELECT
c.CustomerID,
c.FirstName,
c.LastName,
c.EmailAddress,
c.Phone,
c.CompanyName
FROM
SalesLT.Customer c

```

```

JOIN
CustomerPurchaseCounts pc ON c.CustomerID = pc.CustomerID
WHERE
pc.PurchaseCount > (SELECT AVG(PurchaseCount) FROM
CustomerPurchaseCounts);

```

42. Display products that have been ordered more than the average number of times.

```

WITH ProductOrderCounts AS (
SELECT
p.ProductID,
p.Name AS ProductName,
COUNT(od.SalesOrderID) AS OrderCount
FROM
SalesLT.Product p
JOIN
SalesLT.SalesOrderDetail od ON p.ProductID = od.ProductID
GROUP BY
p.ProductID, p.Name
)
SELECT
ProductID,
ProductName,
OrderCount
FROM
ProductOrderCounts
WHERE
OrderCount > (SELECT AVG(OrderCount) FROM ProductOrderCounts);

```

The screenshot shows the Azure Data Studio interface with the following details:

- Header:** dbyash (snyash/dbyash) | Query editor (preview)
- Toolbar:** Login, New Query, Open query, Feedback, Getting started
- Left Sidebar:** Shows the user dbyash (dbadmin) and a note about showing limited object explorer.
- Object Explorer:** Lists Tables, Views, and Stored Procedures.
- Query Editor:**
 - Query 1: WITH ProductOrderCounts AS (...)
 - Run button and other toolbar options: Run, Cancel query, Save query, Export data as, Show only Editor.
- Results:**
 - Results tab: Shows the output of the query.
 - Table Headers: ProductID, ProductName, OrderCount
 - Data Rows:

ProductID	ProductName	OrderCount
712	AWC Logo Cap	9
877	Bike Wash - Dissolver	7

43.Retrieve orders placed by employees who have completed a specific training course.

Data Not Available

44.List employees who have a higher salary than at least one employee in another department.

Data Not Available

45.Display products that have not been ordered in the last 60 days.

Data Not Available

46.Find employees who have the same job title as the employee with the highest salary.

Data Not Available

47.List customers who have placed orders with a total amount greater than the total amount of a specific order.

48.Retrieve products that have been ordered by customers with the same shipping address.

Not Enough Data

```
WITH CustomerShippingAddresses AS (
SELECT
ca.CustomerID,
ca.AddressID,
a.AddressLine1,
a.AddressLine2,
a.City,
a.StateProvince,
a.CountryRegion,
a.PostalCode
FROM
SalesLT.CustomerAddress ca
JOIN
SalesLT.Address a ON ca.AddressID = a.AddressID
)
SELECT
p.ProductID,
p.Name AS ProductName,
od.SalesOrderID,
csa.CustomerID,
csa.AddressID,
```

```
    csa.AddressLine1,  
    csa.AddressLine2,  
    csa.City,  
    csa.StateProvince,  
    csa.CountryRegion,  
    csa.PostalCode  
  FROM  
    SalesLT.Product p  
  JOIN  
    SalesLT.SalesOrderDetail od ON p.ProductID = od.ProductID  
  JOIN  
    SalesLT.SalesOrderHeader oh ON od.SalesOrderID = oh.SalesOrderID  
  JOIN  
    CustomerShippingAddresses csa ON oh.CustomerID = csa.CustomerID  
 WHERE  
  oh.ShipToAddressID IN (  
    SELECT  
      ShipToAddressID  
    FROM  
      SalesLT.SalesOrderHeader  
    GROUP BY  
      ShipToAddressID  
    HAVING  
      COUNT(DISTINCT CustomerID) > 1
```