한양생명과학기술원
HY Institute of Bioscience and Biotechnology

BI/컴퓨팅 서버 자율사용자 이용 가이드

2025.10

목차

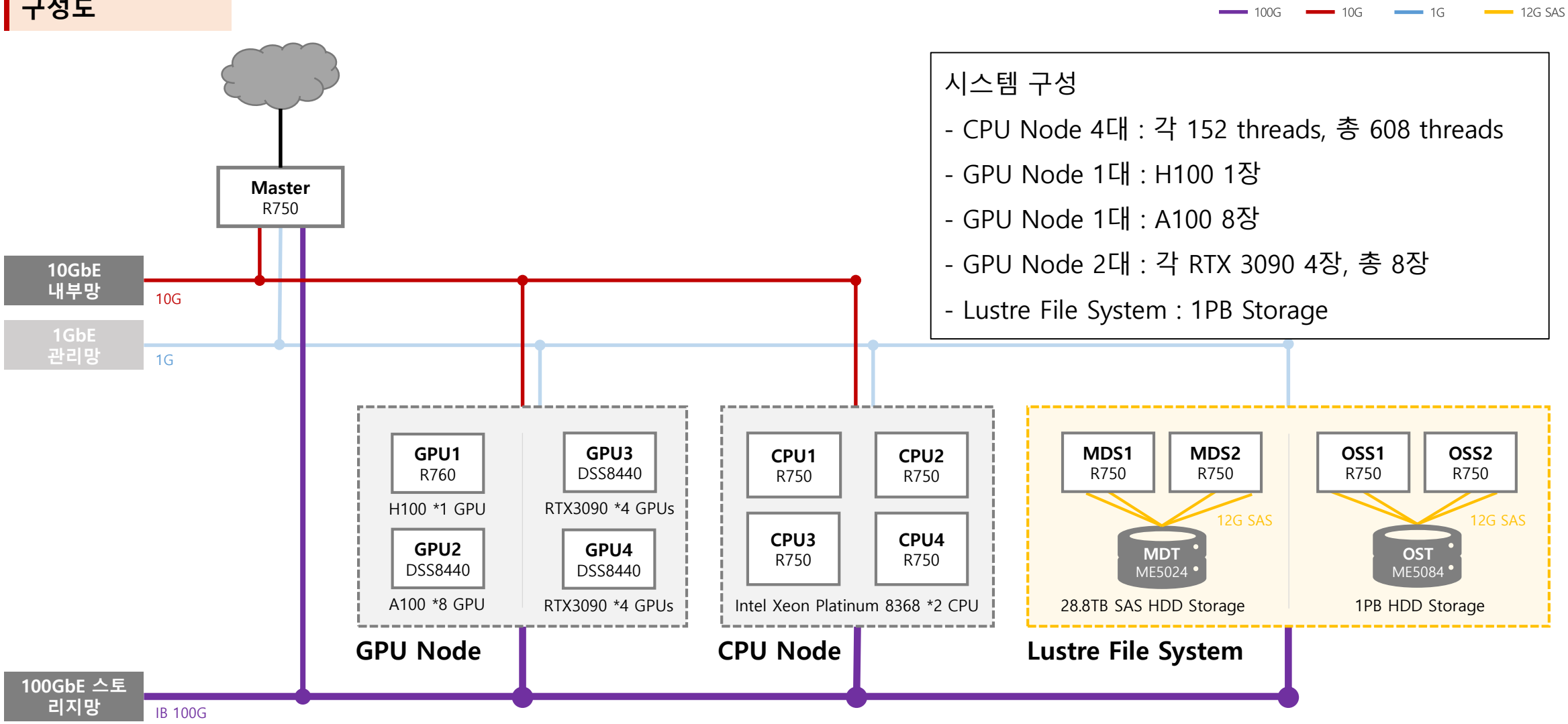
Index

1. BI/컴퓨팅 코어 클러스터 구성 - 3
2. 접속방법 - 5
3. 비밀번호 변경 - 24
4. SLURM 활용 - 26
5. Environment Module - 37
6. Anaconda - 40
7. 한양 VPN - 47

1. 클러스터 구성

1. BI/컴퓨팅 코어 클러스터 구성

구성도



시스템 구성

- CPU Node 4대 : 각 152 threads, 총 608 threads
- GPU Node 1대 : H100 1장
- GPU Node 1대 : A100 8장
- GPU Node 2대 : 각 RTX 3090 4장, 총 8장
- Lustre File System : 1PB Storage

한양생명과학기술원

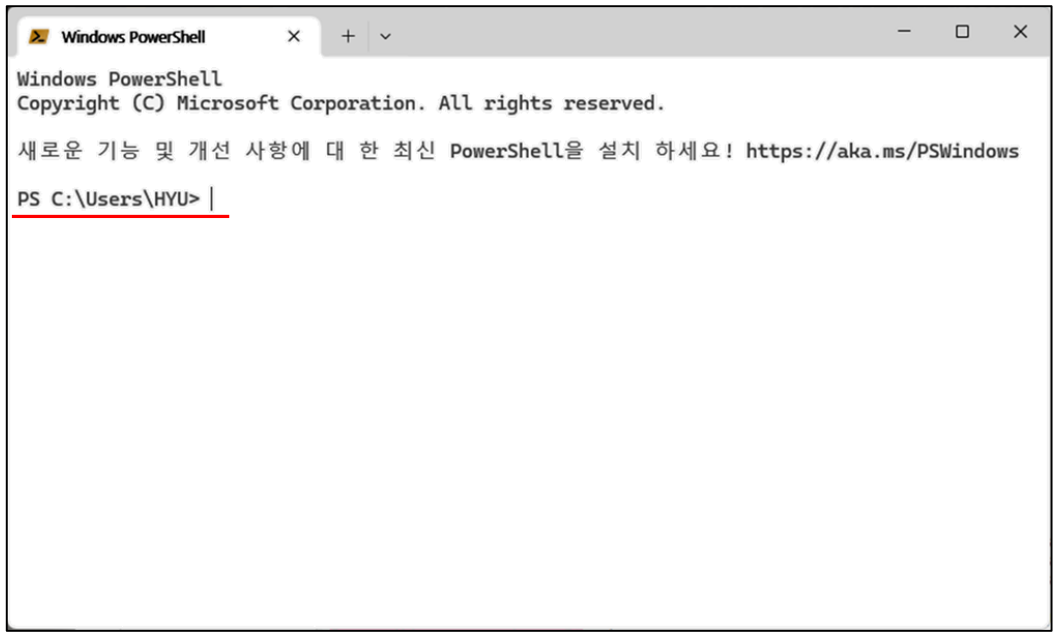
2. 접속 방법

한양생명과학기술원
HY Institute of Bioscience and Biotechnology

2. 접속 방법 (1. 기본 터미널)

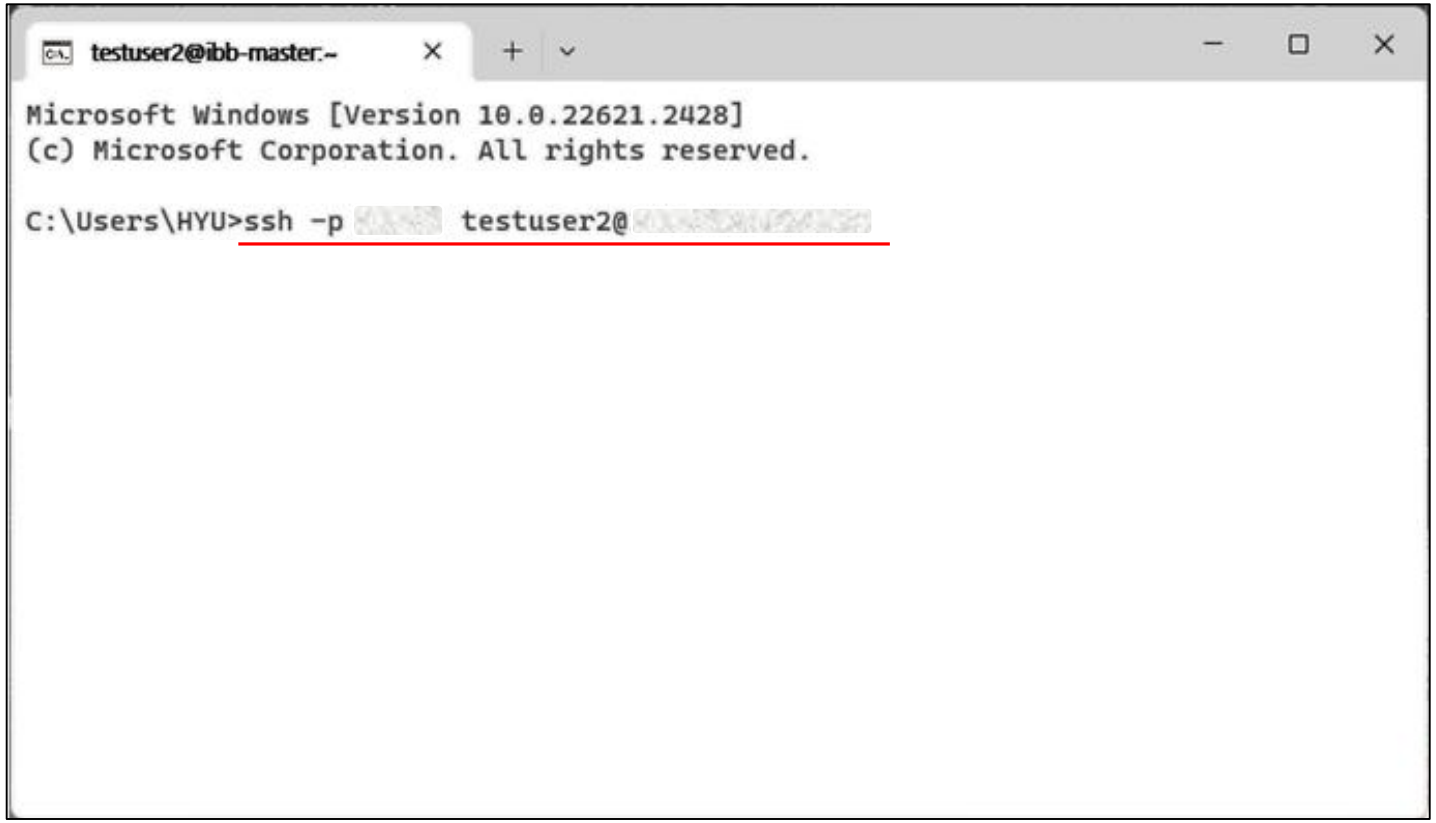
※ 클러스터 서버에 대한 접근은 ssh 프로토콜을 사용하며 mac OS 나 Windows 11 같은 경우에는
기본 터미널에서 ssh 프로토콜을 지원하기때문에 별도 프로그램을 설치하지 않아도 서버에 접속할 수 있습니다.





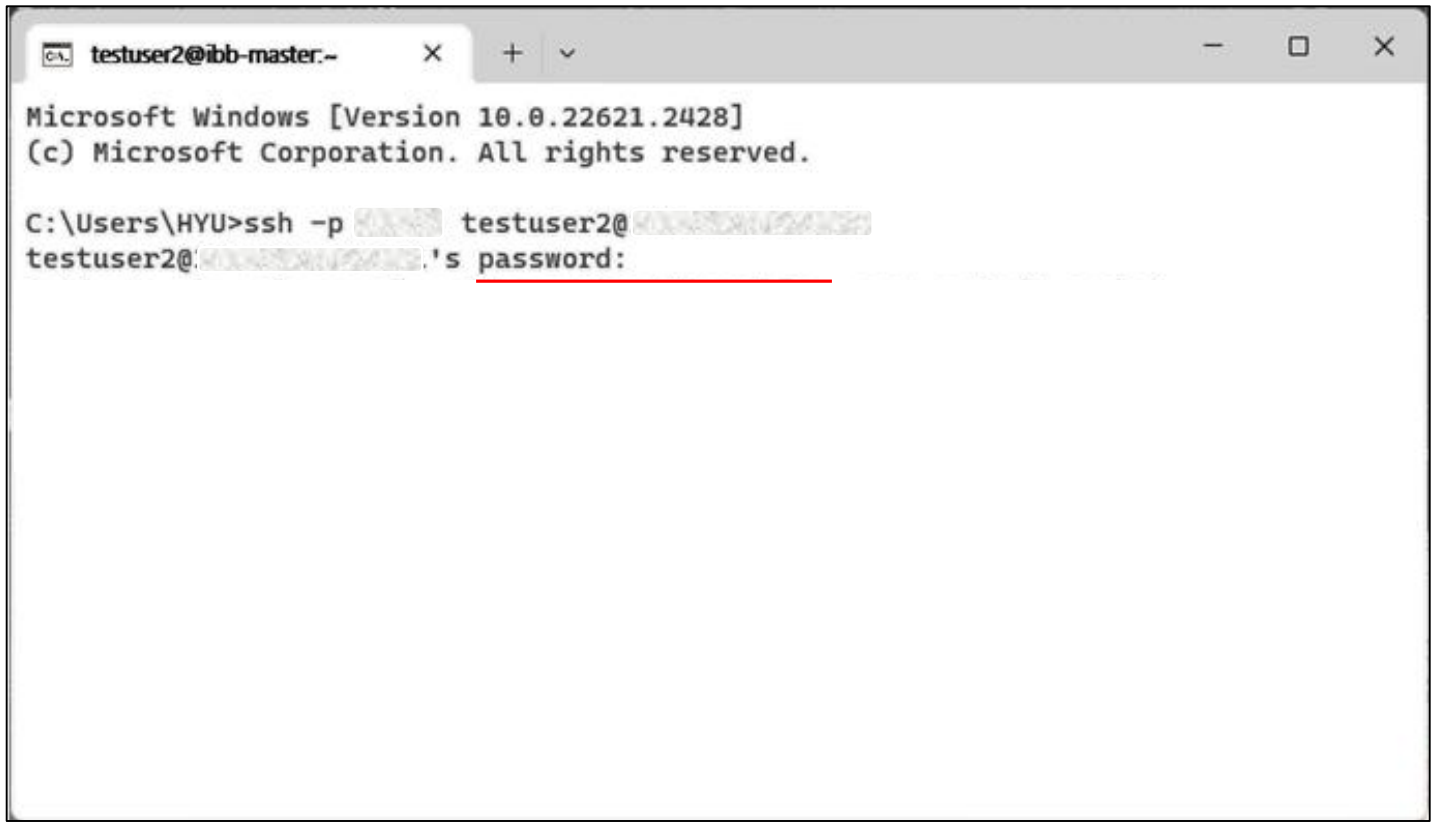
1. 시작메뉴에 '터미널' 혹은 'terminal'을 검색하여 기본 터미널을 실행합니다.
(초기 화면은 로컬 터미널 환경으로 구성되어 있습니다.)

2. 접속 방법 (1. 기본 터미널)



2. <ssh -p [포트번호] **userID**@[서버IP주소]> 와 같이 ssh 명령어를 입력하여 클러스터 서버에 접속합니다.
   userID 부분에 사용자 계정 아이디를 입력합니다.
   (IBB 클러스터 서버의 IP 주소와 포트 번호는 보안을 위하여 서버 사용자에게 개별 안내됩니다.)

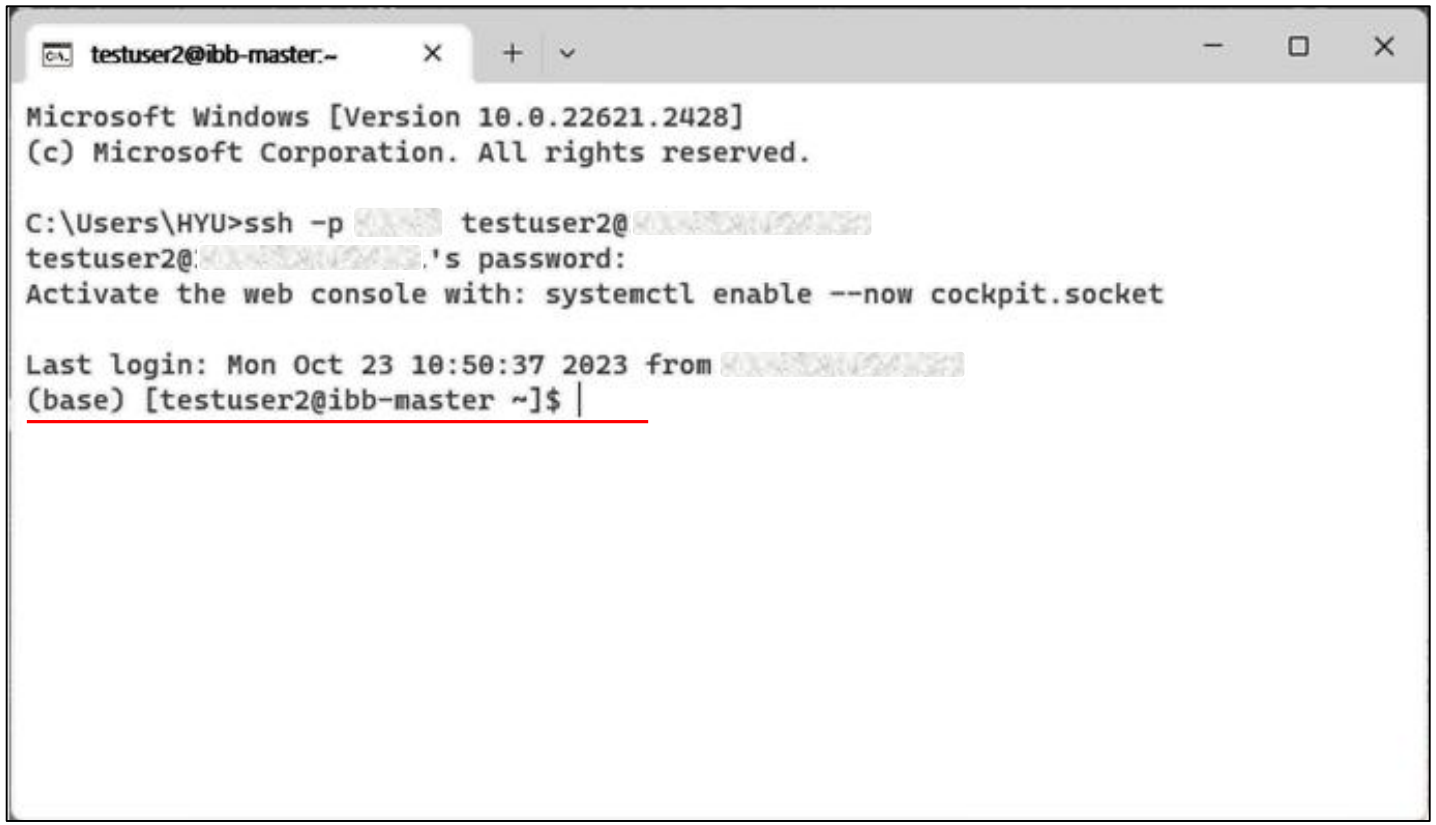
2. 접속 방법 (1. 기본 터미널)



3. password 를 입력하라는 메시지가 출력되면 비밀번호를 입력하여 계정에 로그인 합니다.
(비밀번호는 화면에 출력되지 않습니다.)

한양생명과학기술원

2. 접속 방법 (1. 기본 터미널)



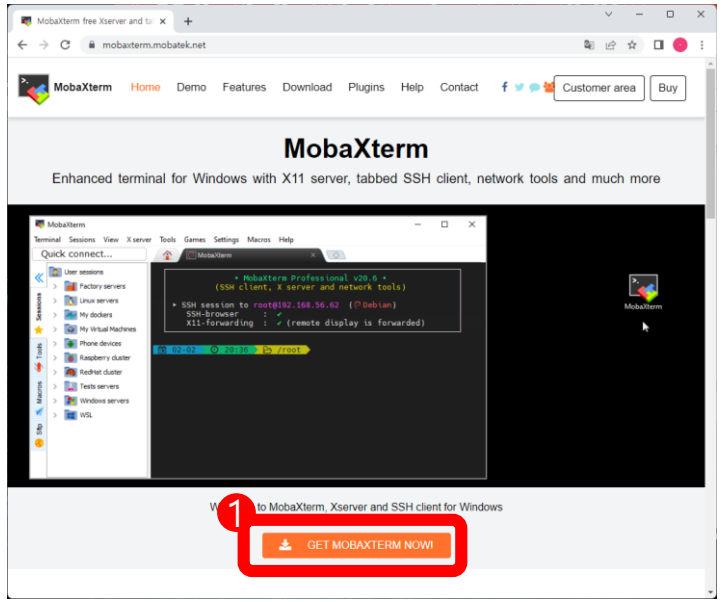
4. 계정 로그인이 정상적으로 완료되면 서버 터미널을 이용할 수 있습니다.

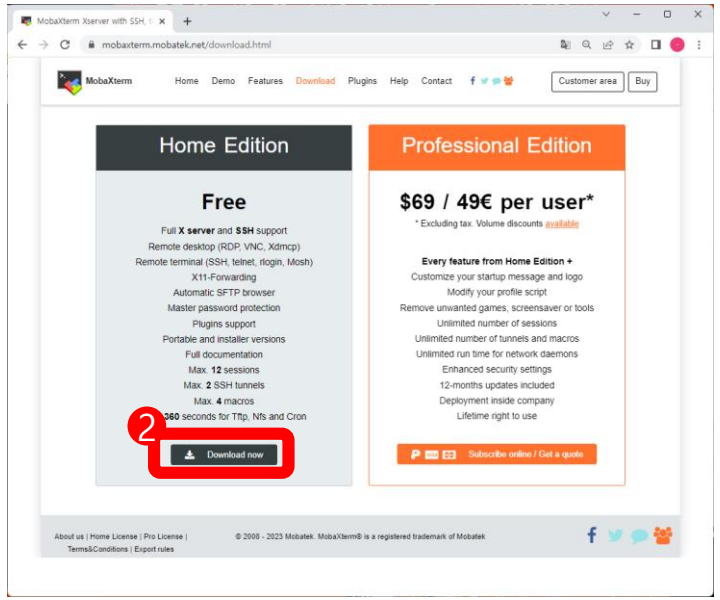
한양생명과학기술원
HY Institute of Bioscience and Biotechnology

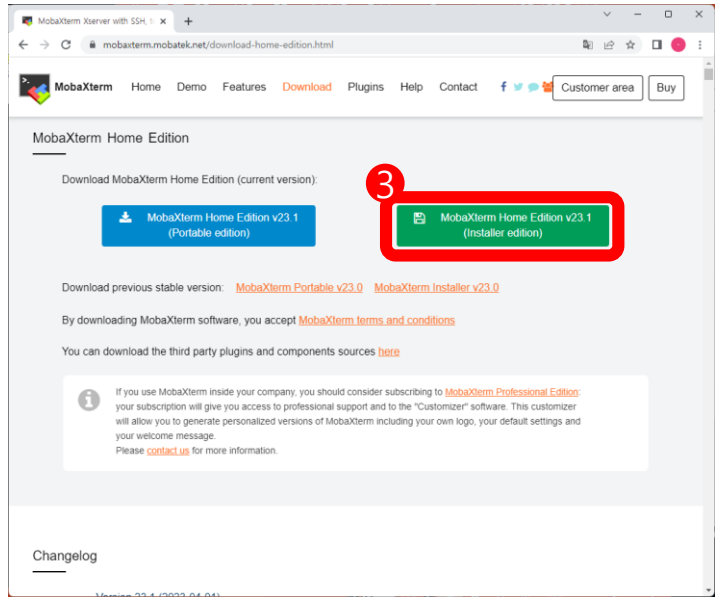
2. 접속 방법 (2. MobaXterm)

※ 서버 접속에는 MobaXterm, Visual Studio Code, PuTTY 와 같이 다양한 프로그램을 활용할 수 있습니다.
다음은 MobaXterm을 통해 서버에 접근하는 방법입니다. (Mac OS 환경에서는 MobaXterm을 지원하지 않아 사용 불가합니다.)

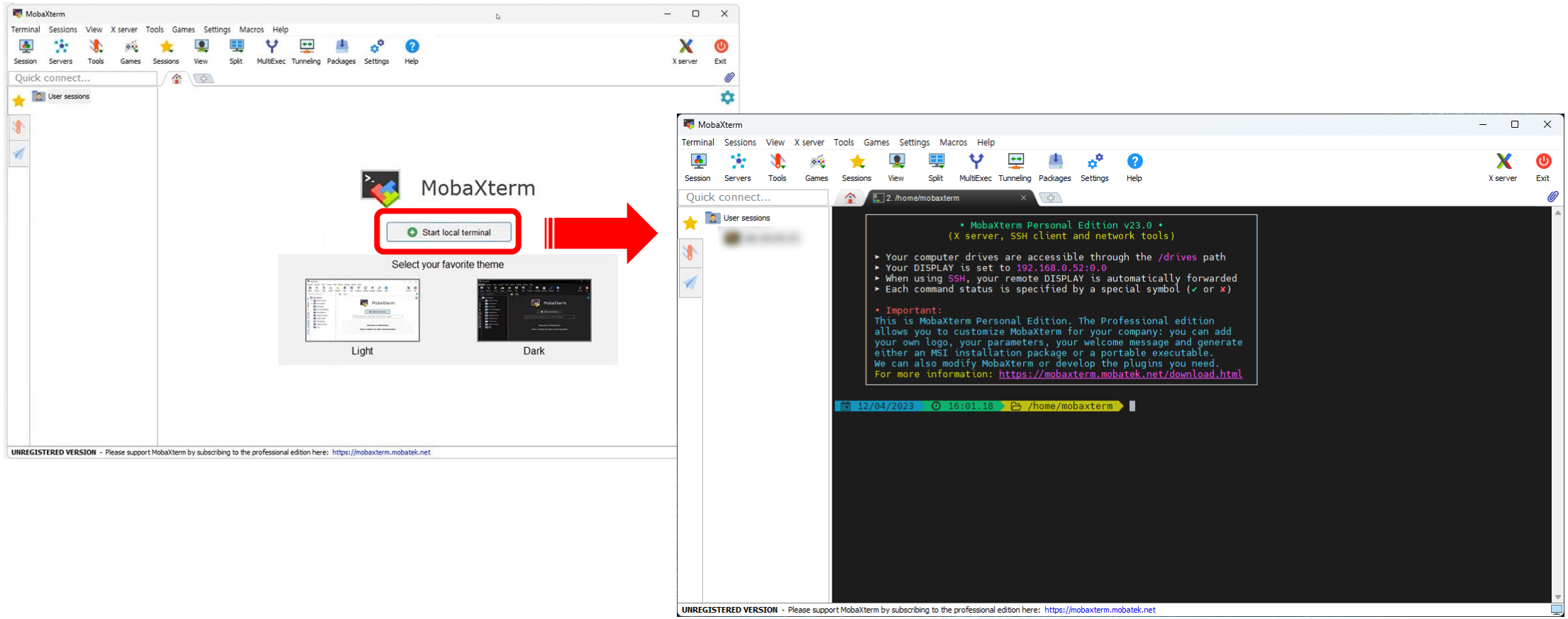
1. 프로그램 설치를 위해 MobaXterm 홈페이지(https://mobaxterm.mobatek.net/)에 접속 후,
아래와 같은 과정으로 설치 파일을 다운로드하여 MobaXterm 설치를 진행합니다.







2. 접속 방법 (2. MobaXterm)

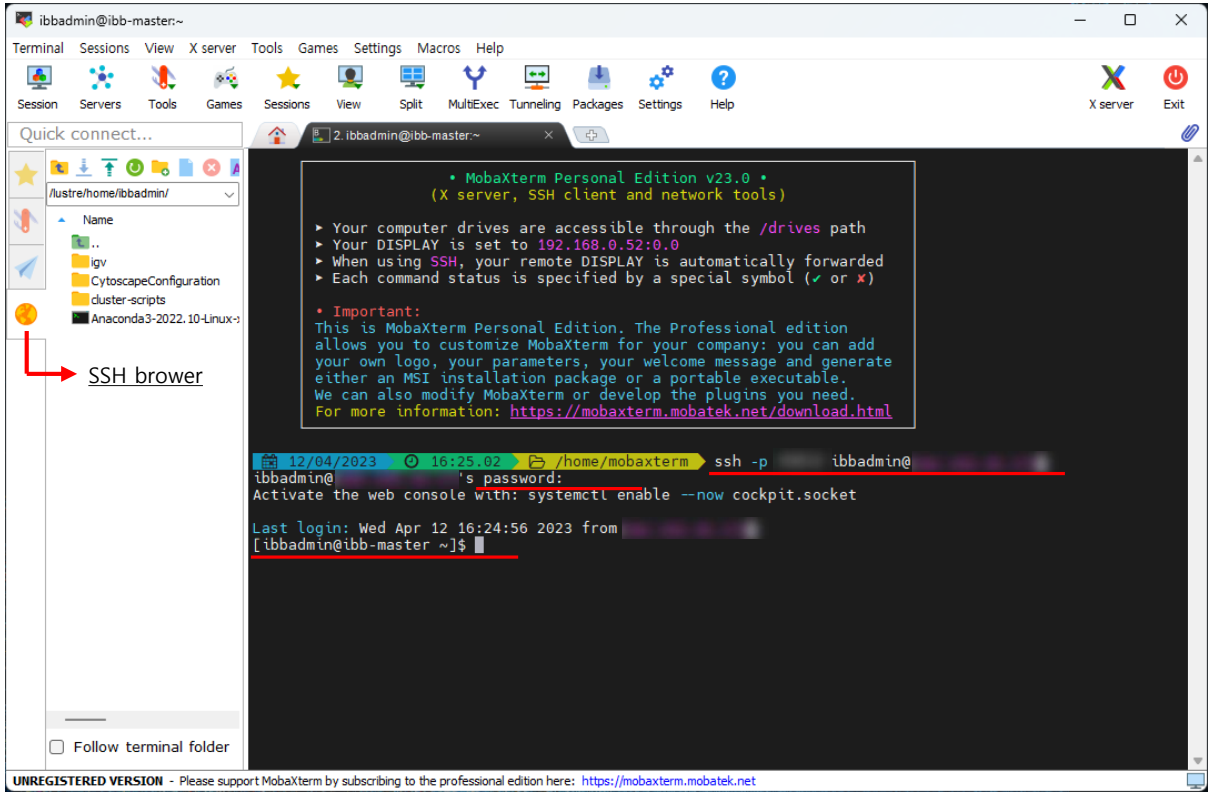


2. MobaXterm을 실행하면 위와 같은 메인 화면이 나타나며

'Start local terminal' 버튼을 클릭하여 로컬 터미널 환경을 시작할 수 있습니다.

한양생명과학기술원

2. 접속 방법 (2. MobaXterm)



3. <ssh -p [포트번호] **userID**@[서버IP주소]> 와 같이 ssh 명령어를 입력 후 비밀번호를 입력하여 계정에 로그인 합니다.
(IBB 클러스터 서버의 IP 주소와 포트 번호는
보안을 위하여 서버 사용자에게 개별 안내됩니다.)

※ MobaXterm에서는 SSH brower를 통해서 Windows와 같은 그래픽 모드로 서버 내 파일을 확인할 수 있습니다.

또한 간단한 파일들은 SSH brower를 통해서
드래그 앤 드롭으로 서버에 업로드하거나
로컬 환경으로 다운로드 받을 수 있습니다.

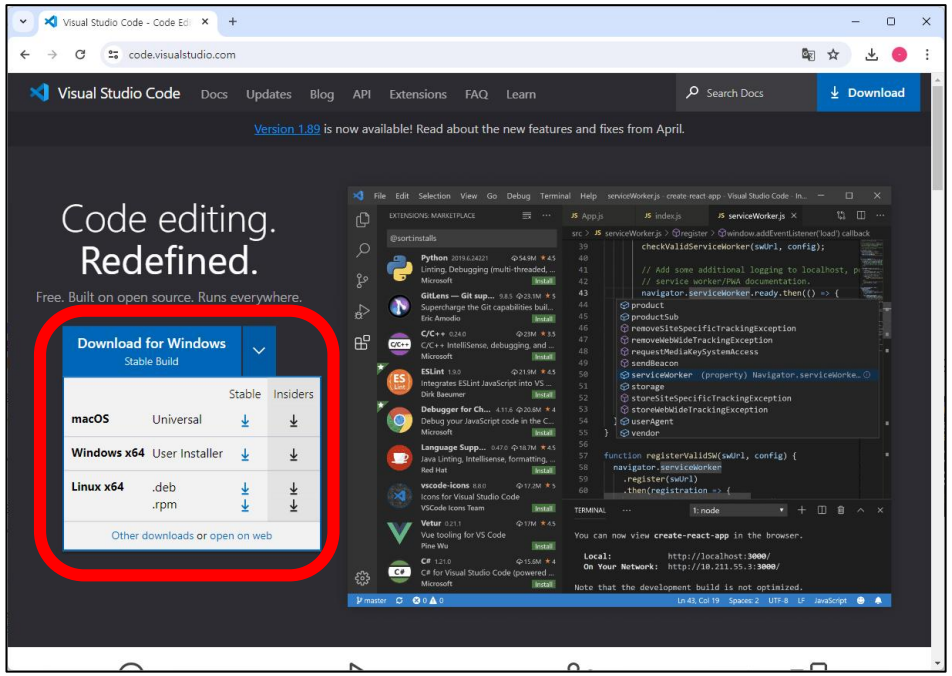
이외에 접속 정보를 저장하거나 멀티 TAB을 지원하는 등
다양한 편의기능을 제공합니다.

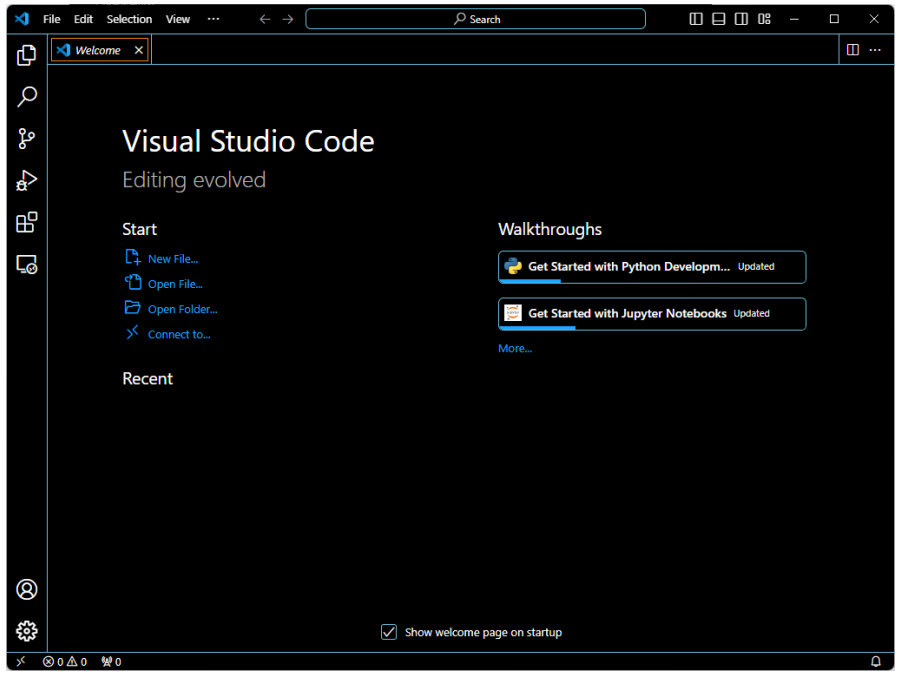
한양생명과학기술원

2. 접속 방법 (3. Visual Studio Code)

※ 서버 접속에는 MobaXterm, Visual Studio Code, PuTTY 와 같이 다양한 프로그램을 활용할 수 있습니다.
다음은 Visual Studio Code를 통해 서버에 접근하는 방법입니다.

1. 프로그램 설치를 위해 Visual Studio Code 홈페이지(https://code.visualstudio.com/)에 접속 후,
사용중인 OS 환경에 맞는 설치 파일을 다운로드하여 Visual Studio Code 설치를 진행합니다.

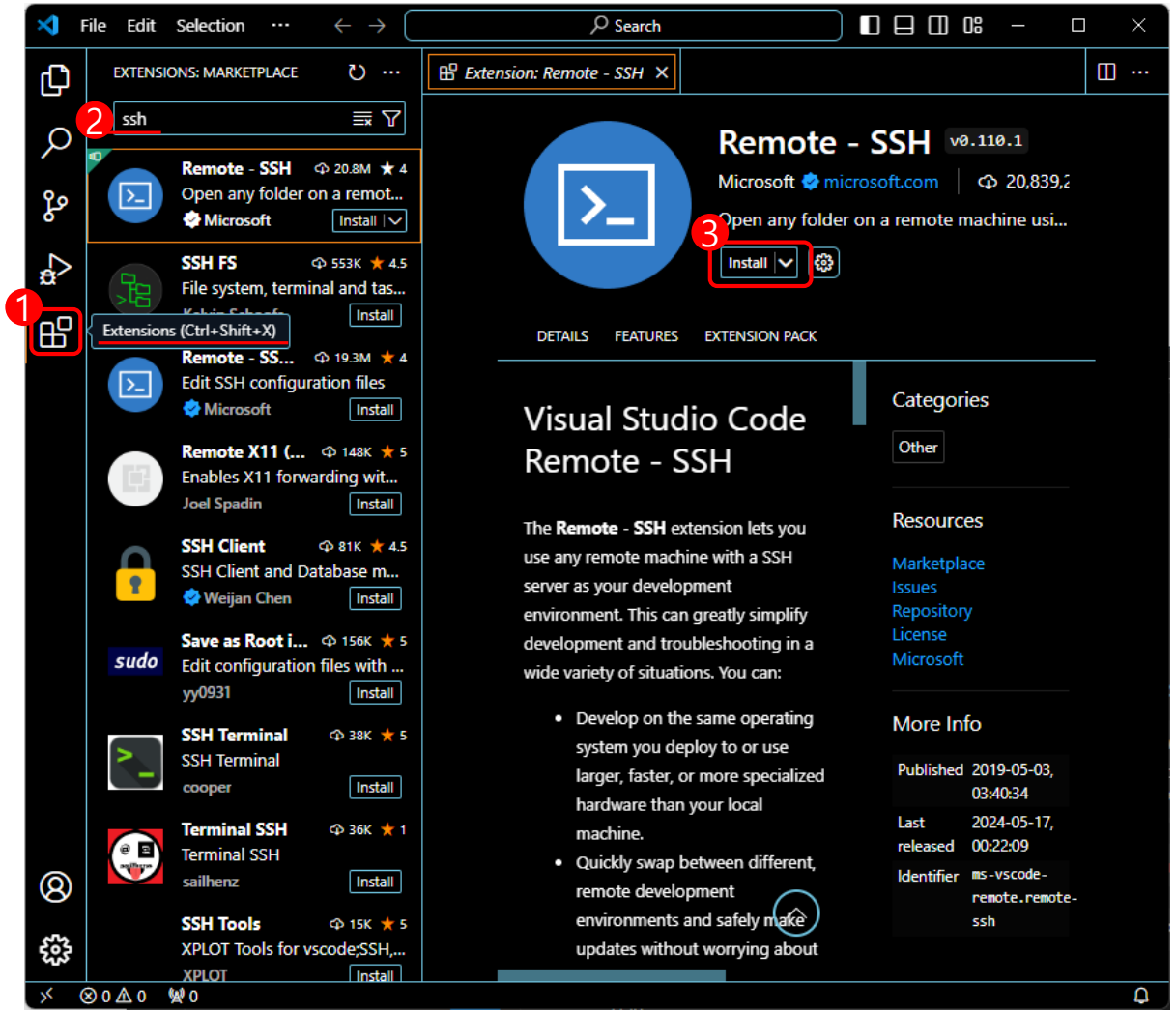




Visual Studio Code
실행 화면

한양생명과학기술원

2. 접속 방법 (3. Visual Studio Code)

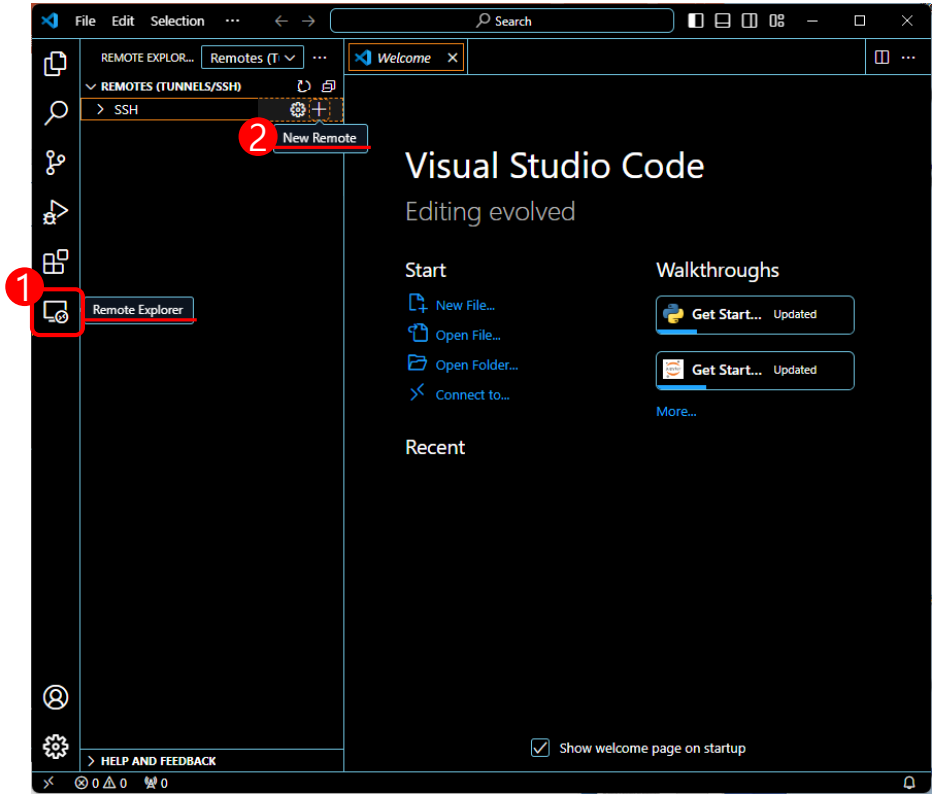


2. Visual Studio Code는 다양한 확장 기능을 제공합니다.
   좌측의 Extensions(확장) 탭을 선택하고 'ssh'를 검색하여
   'Remote - SSH' 확장 기능을 설치합니다.

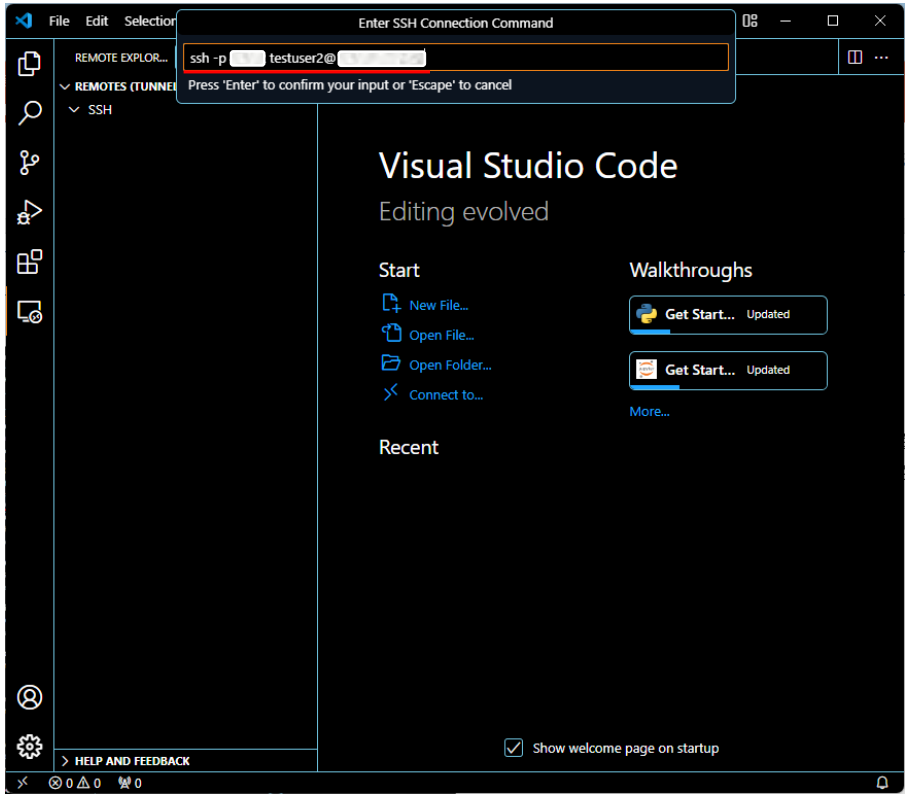
※ SSH 접속 이외에도 다양한 확장 기능을 설치하여
   활용할 수 있습니다.

한양생명과학기술원

2. 접속 방법 (3. Visual Studio Code)



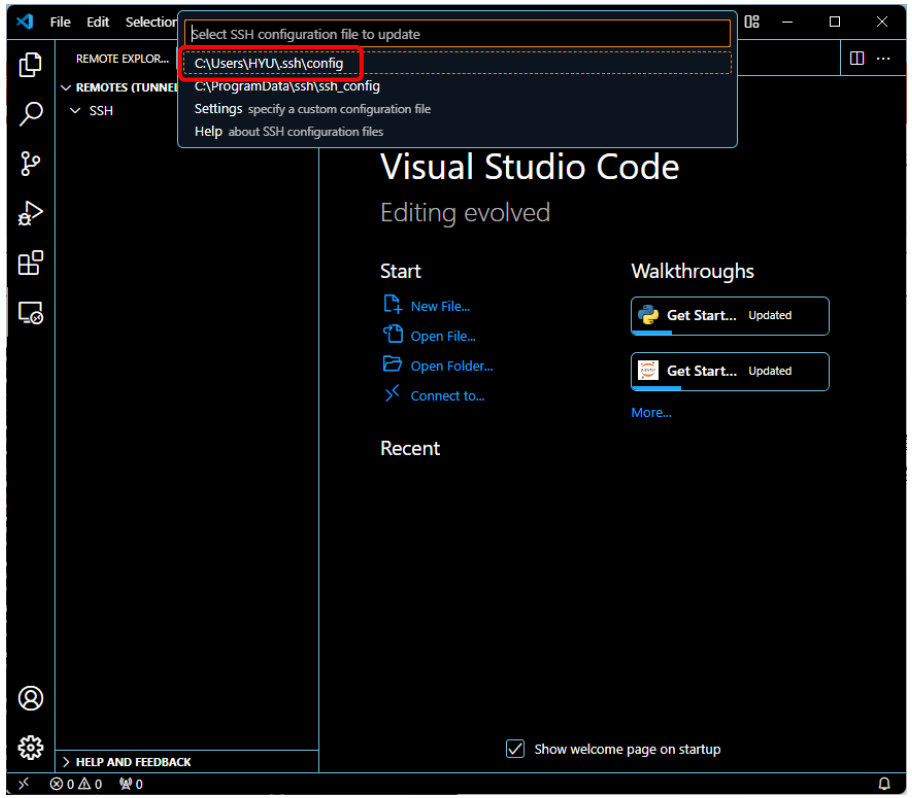
3. 'Remote - SSH' 확장 기능이 설치되면 좌측에
'Remote Explorer' 탭이 추가됩니다.
해당 탭을 선택하고 'New Remote' 버튼을 클릭합니다.



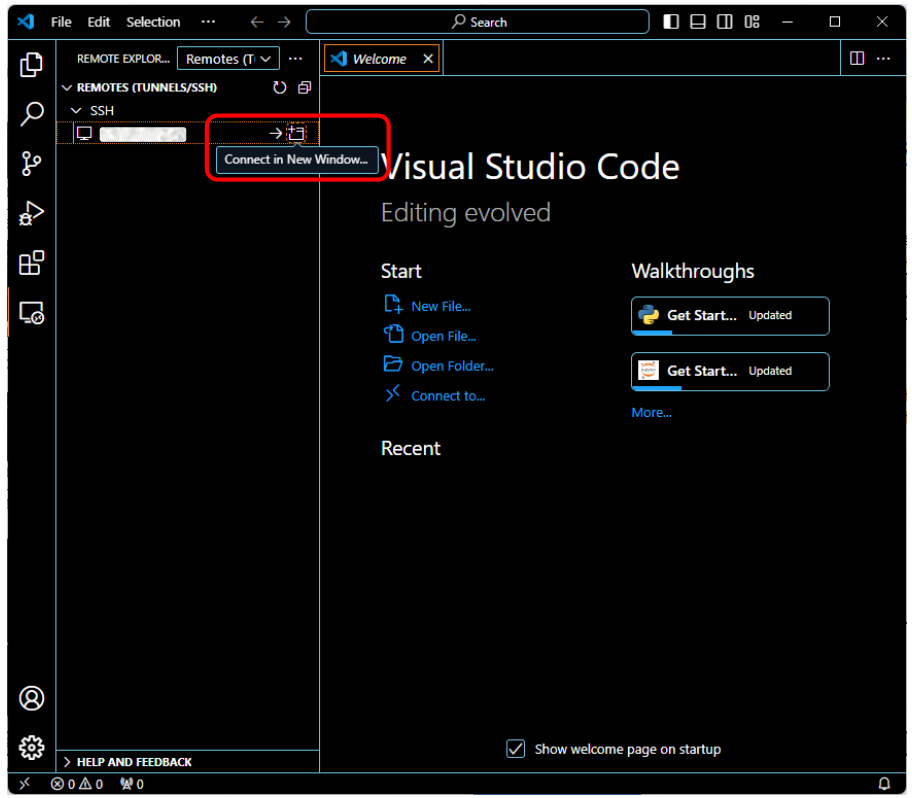
4. <ssh -p [포트번호] **userID**@[서버IP주소]> 와 같이
ssh 명령어를 입력하여 ssh config 파일을 생성합니다.
(IBB 클러스터 서버의 IP 주소와 포트 번호는
보안을 위하여 서버 사용자에게 개별 안내됩니다.)

한양생명과학기술원

2. 접속 방법 (3. Visual Studio Code)



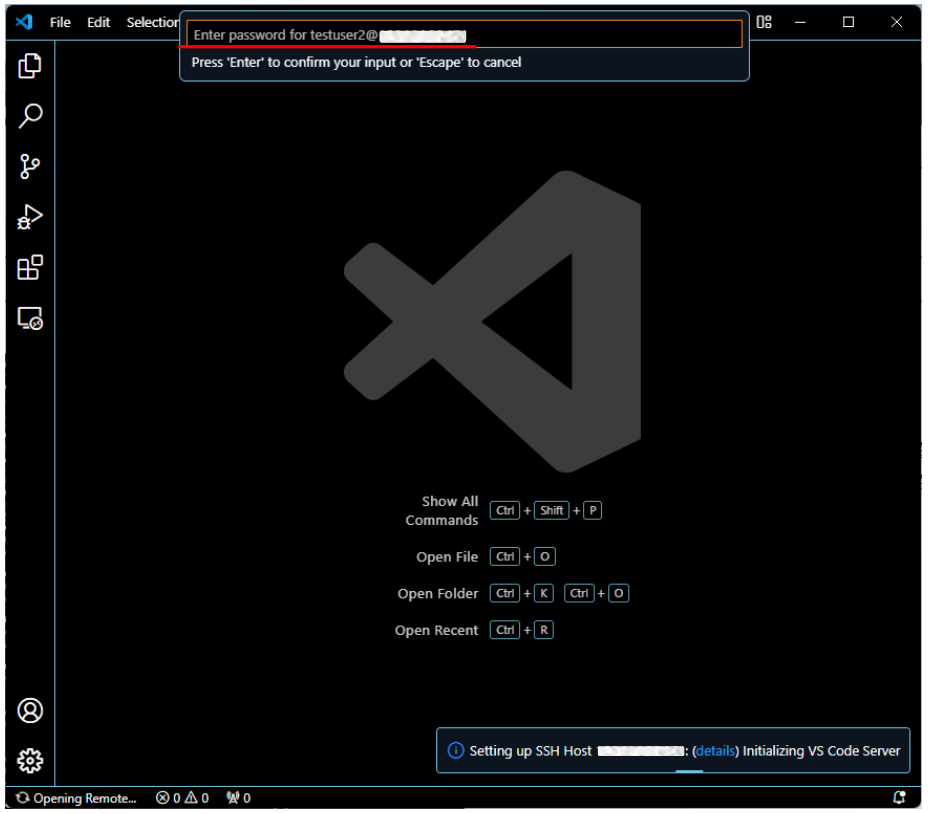
5. 로컬 PC에 SSH 정보를 저장할 config 파일 위치를 선택합니다.



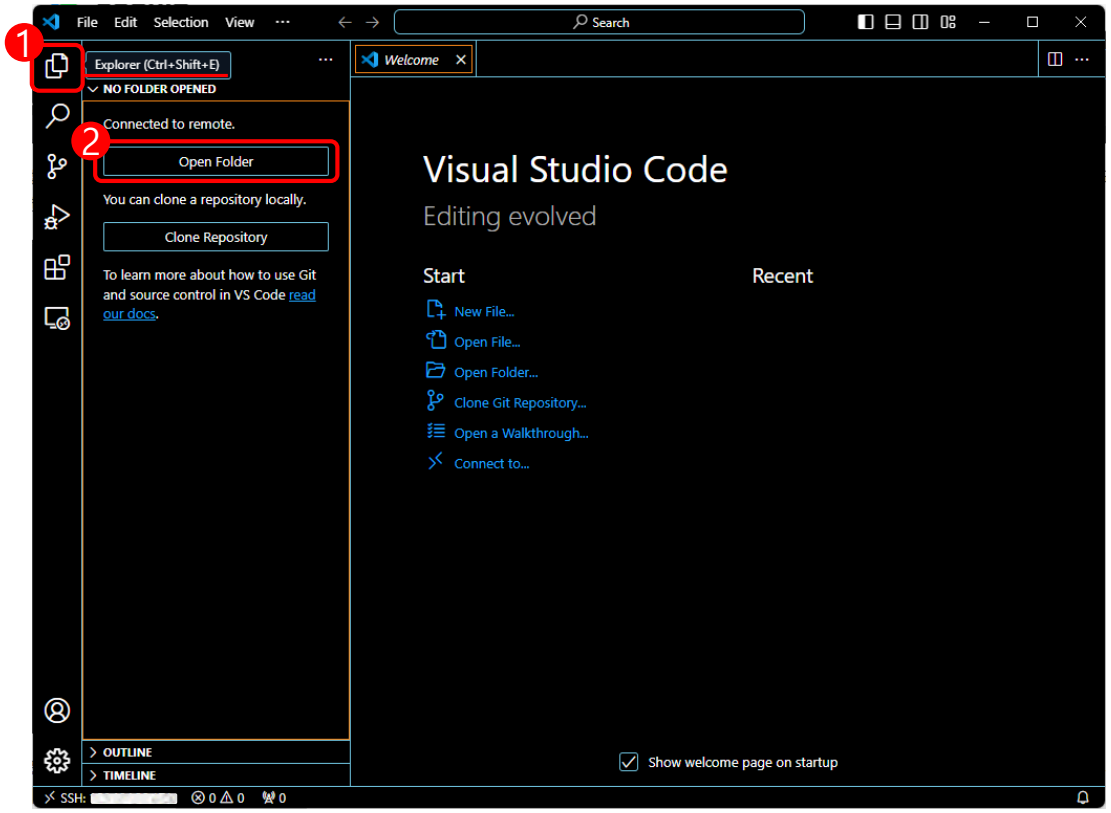
6. 입력한 서버 IP 주소가 추가되며
연결 버튼을 클릭하여 서버에 접속합니다.

한양생명과학기술원

2. 접속 방법 (3. Visual Studio Code)



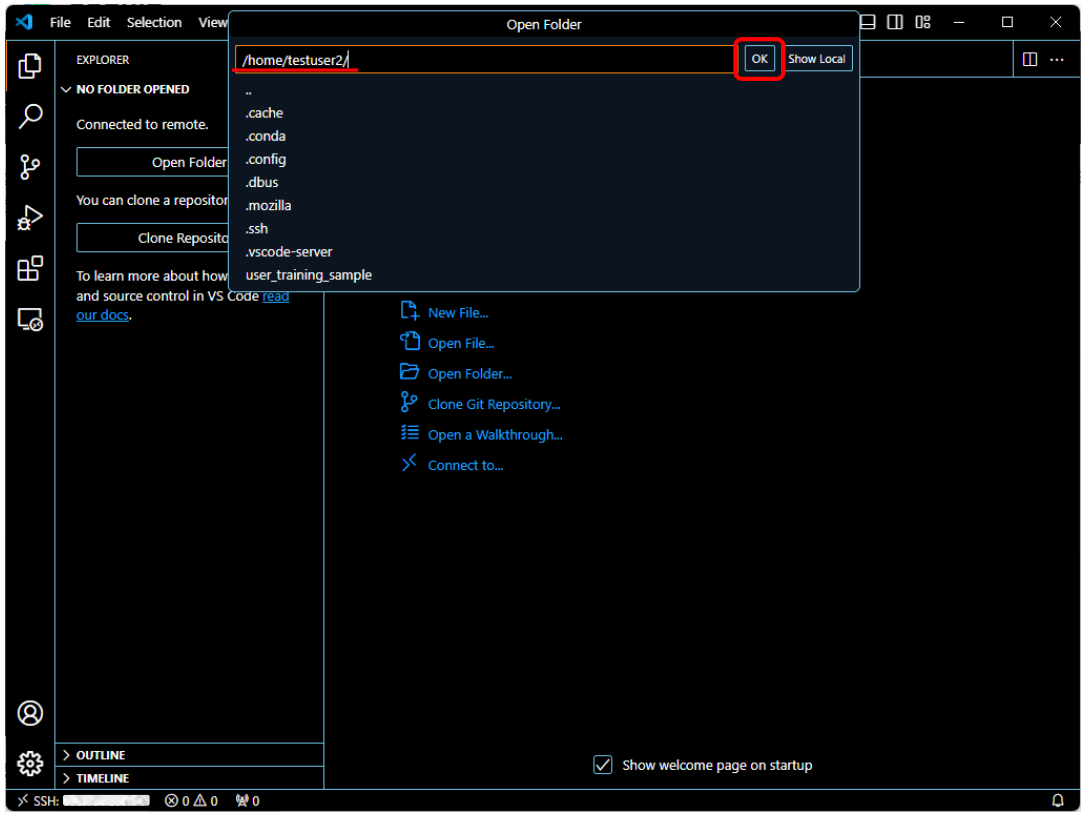
7. 서버 접속 창에서 계정 비밀번호를 입력합니다.
(초기 접속 시 운영체제를 선택하라는 메시지가 출력되면 서버 OS 환경인 Linux를 선택합니다.)



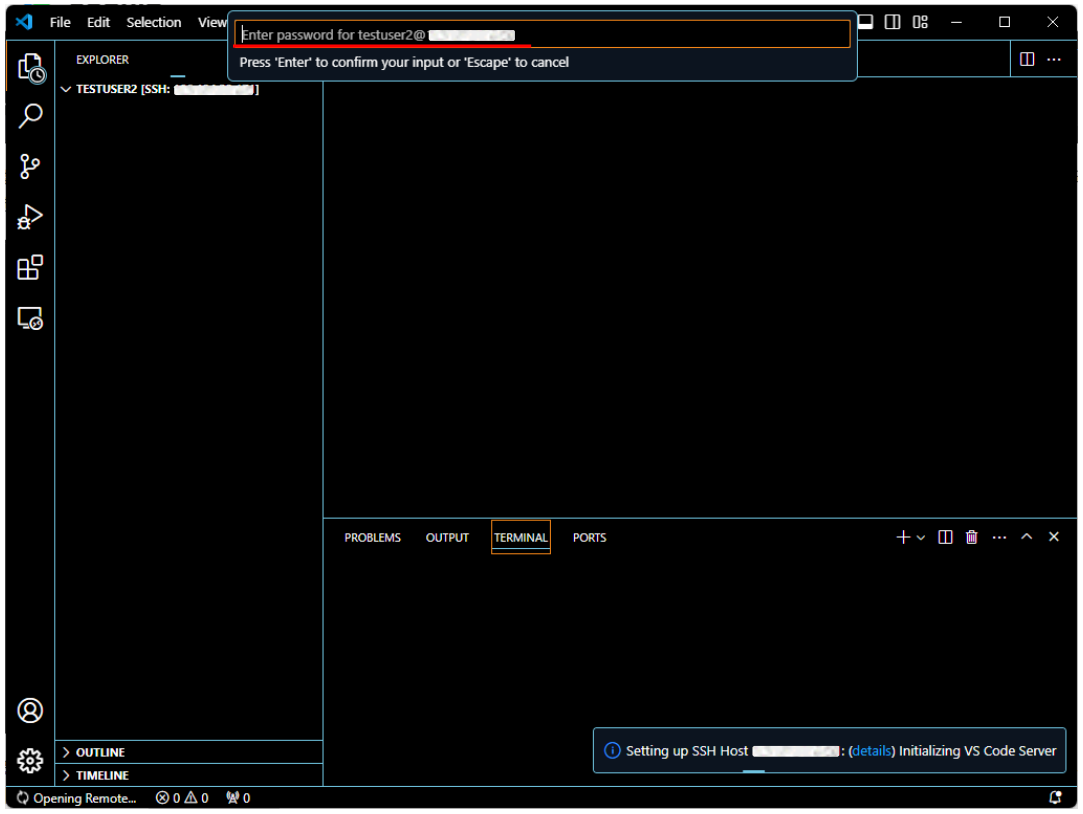
8. 연결이 완료되면 좌측의 'Explorer' 탭을 선택하고 'Open Folder' 버튼을 클릭합니다.

한양생명과학기술원

2. 접속 방법 (3. Visual Studio Code)



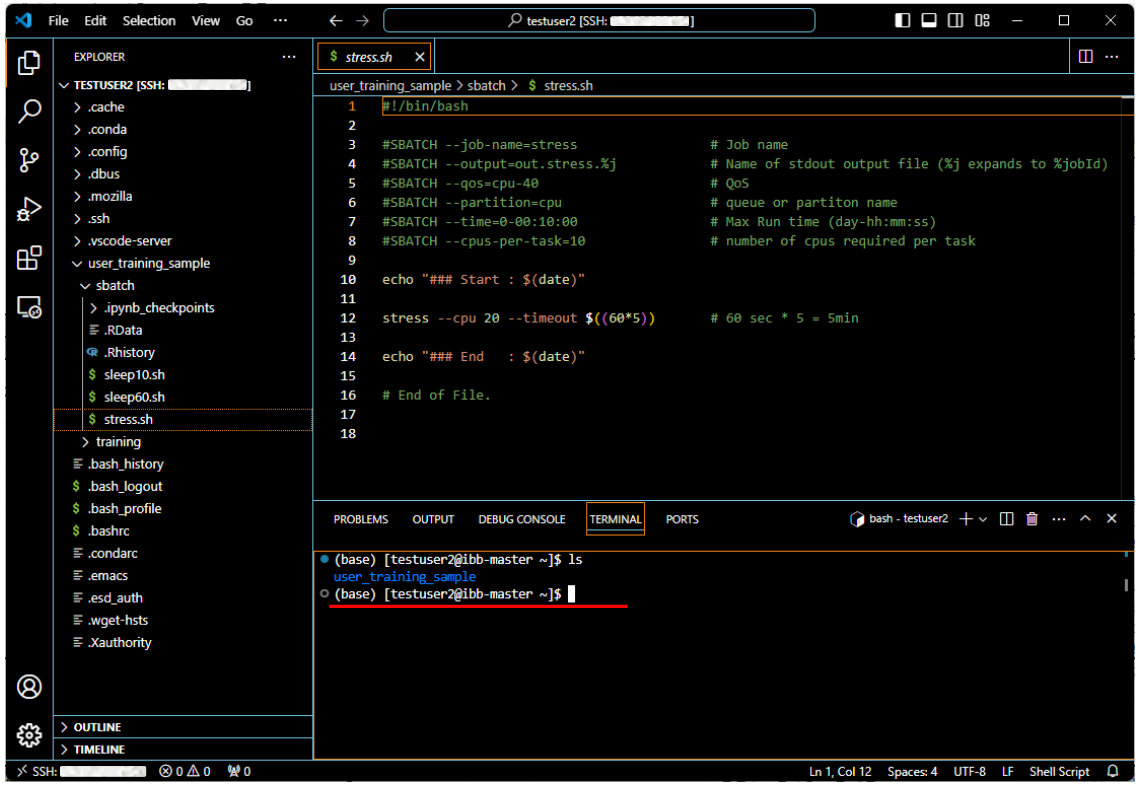
9. 기본적으로 지정되어 있는 홈 디렉토리를 선택합니다.



10. 계정 비밀번호를 입력하여 다시 서버에 접속합니다.

한양생명과학기술원

2. 접속 방법 (3. Visual Studio Code)



11. 계정 로그인이 정상적으로 완료되면
서버 터미널을 이용할 수 있습니다.
(상단 메뉴의 'View > Terminal' 을 선택하거나
[Ctrl+`] 키를 통해 터미널을 활성화합니다.)

※ Visual Studio Code에서는 좌측 'Explorer' 탭을 통해
서버 내 파일 목록을 확인할 수 있습니다.

또한 코드 파일을 열어 내용을 수정하면서
동시에 터미널 환경을 사용 할 수 있습니다.

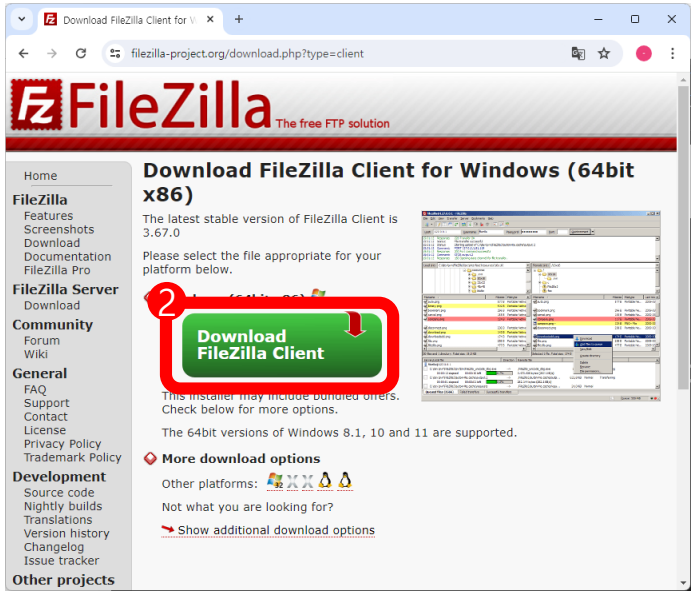
한양생명과학기술원
HY Institute of Bioscience and Biotechnology

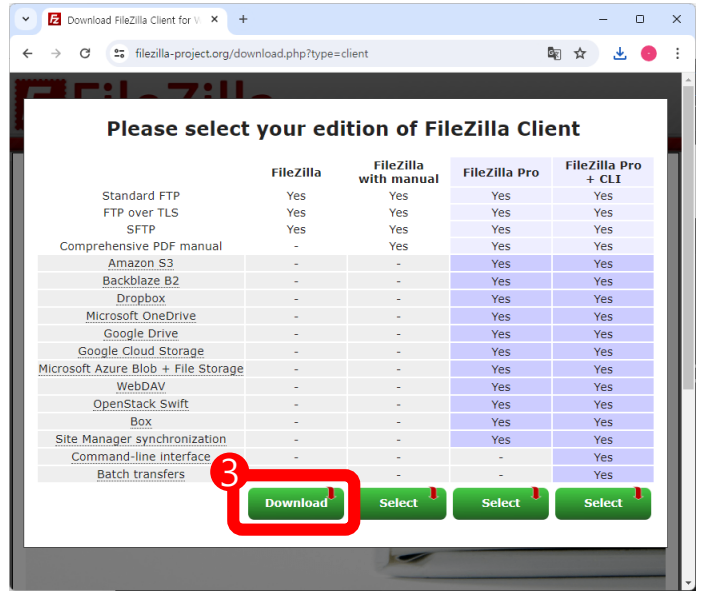
2. 접속 방법 (4. FileZilla)

※ 다음은 PC와 웹사이트 또는 웹 서버 간 파일 전송에 사용되는
FTP(File Transfer Protocol) 프로그램 중 하나인 FileZilla에 대한 사용 방법입니다.

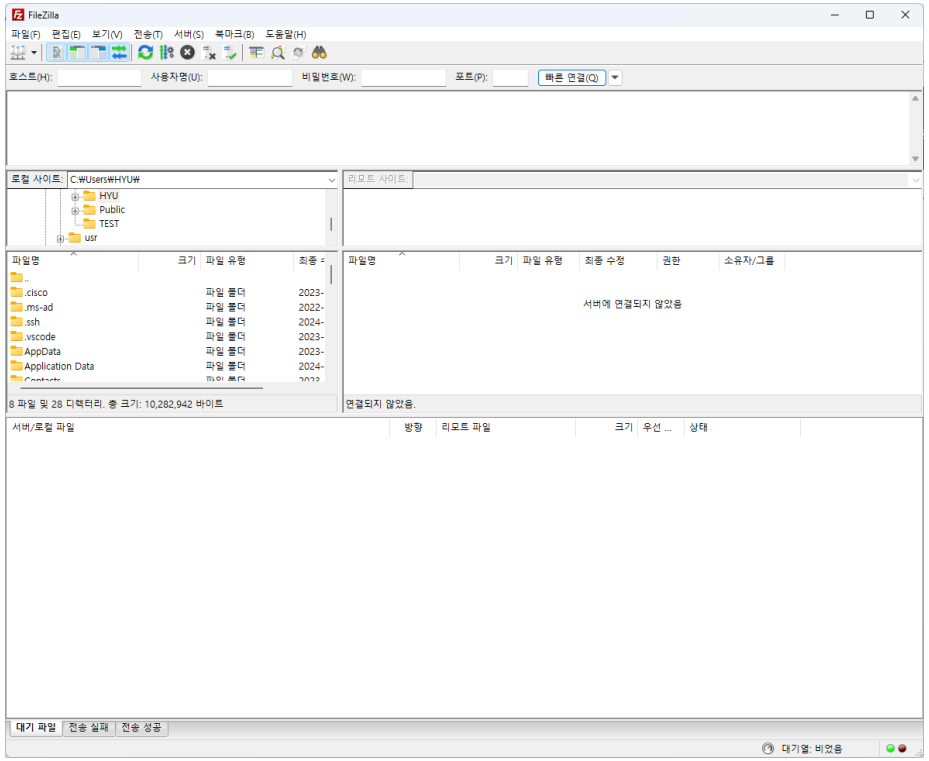
1. 프로그램 설치를 위해 FileZilla 홈페이지(https://filezilla-project.org/)에 접속 후,
아래와 같은 과정으로 사용중인 OS 환경에 맞는 설치 파일을 다운로드하여 FileZilla 설치를 진행합니다.



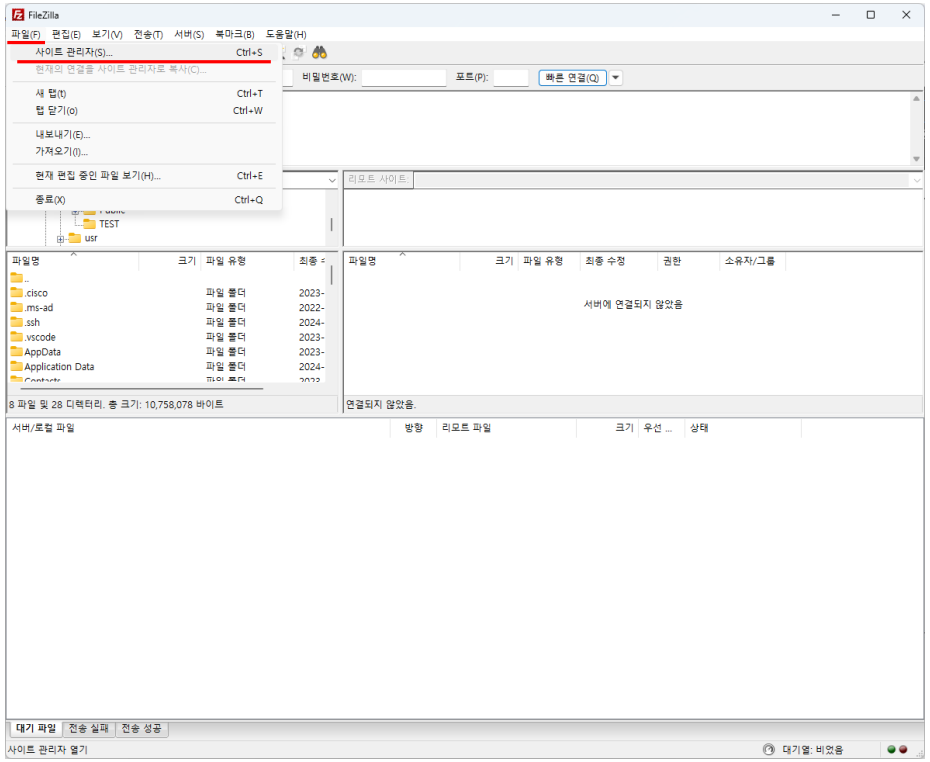




2. 접속 방법 (4. FileZilla)



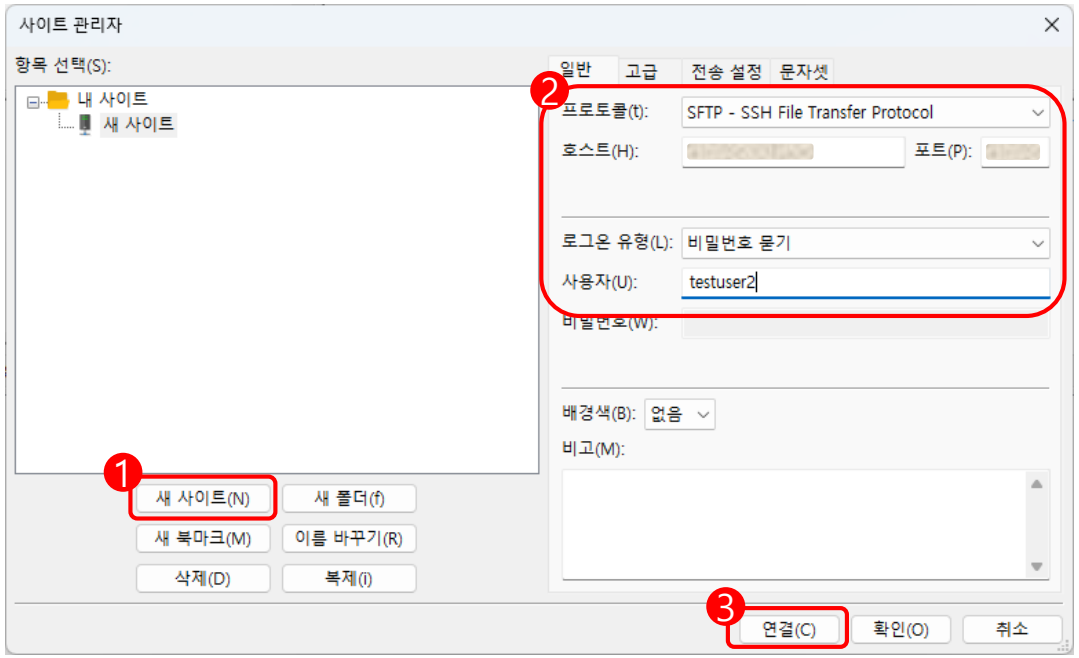
2. FileZilla를 실행하면 위와 같은 화면이 나타납니다.
   좌측에 로컬 PC 환경이, 우측에 연결된 서버 환경이
   하단에 파일 전송 현황이 표시됩니다.

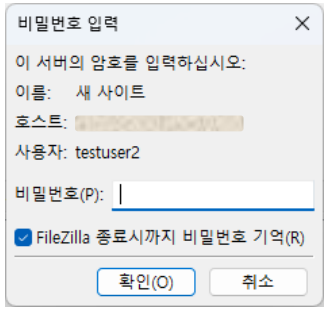


3. '파일 > 사이트 관리자' 메뉴를 선택합니다.

한양생명과학기술원
HY Institute of Bioscience and Biotechnology

2. 접속 방법 (4. FileZilla)





4. '새 사이트(N)' 버튼을 클릭하여 생성합니다.

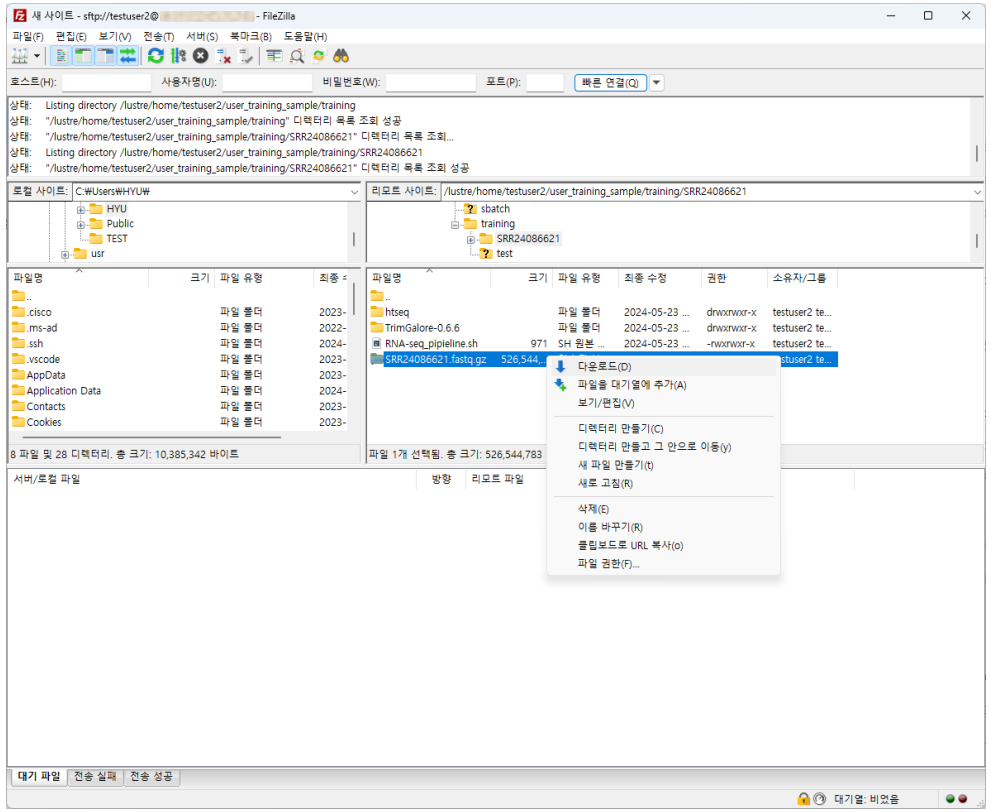
5. 프로토콜(t)은 'SFTP - SSH File Transfer Protocol'을 선택합니다.
   호스트(H)에 접속할 '서버 IP 주소'를 입력합니다.
   포트(P)에 접속할 '포트번호'를 입력합니다.
   로그온 유형은 '비밀번호 묻기'를 선택합니다.
   사용자에 '계정 아이디'를 입력합니다.

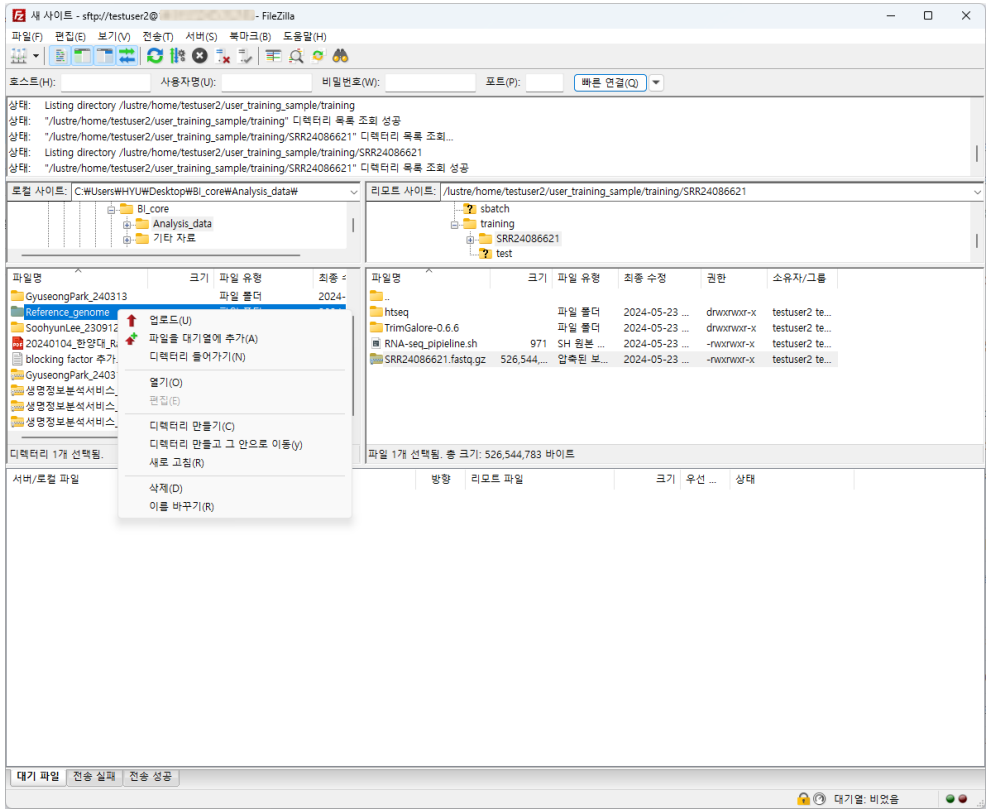
6. 연결 버튼을 클릭하여 서버에 연결합니다.

7. 비밀번호를 입력하라는 창이 뜨면
   계정 비밀번호를 입력합니다.

한양생명과학기술원

2. 접속 방법 (4. FileZilla)





8. 서버의 파일들을 로컬 PC로 다운로드 받거나, 로컬 PC의 파일들을 서버로 업로드 할 수 있습니다.

※ 대용량의 파일이나 다수의 파일들을 손쉽게 전송할 수 있으며 하단 대기열을 통해 파일 전송 상태를 확인 할 수 있습니다.

한양생명과학기술원
HY Institute of Bioscience and Biotechnology

3. 비밀번호 변경

3. 비밀번호 변경

다음과 같은 과정으로 새로운 비밀번호 설정이 가능합니다. (변경없이 초기 설정된 비밀번호 그대로 사용 가능합니다.)

※ 비밀번호는 1개 이상의 숫자, 대문자, 소문자, 특수문자가 포함되고 최소 9자리 이상이어야 하며
동일한 유형의 문자는 연속적으로 3회 까지만 입력 가능합니다. (가능 : asd , 135 / 불가 : asdf , 1357)

```
(base) [testuser2@ibb-master ~]$ passwd
Changing password for user testuser2.
Current password:
New password:
BAD PASSWORD: The password contains more than 3 characters of the same class consecutively
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

1. <passwd> 명령어를 입력하면 비밀번호 변경 프로세스가 시작됩니다.

2. Current password: 현재 비밀번호를 입력합니다. (비밀번호는 화면에 출력되지 않습니다.)

3. New password: 새로운 비밀번호를 입력합니다. (조건에 부합하지 않을 시 BAD PASSWORD 메시지가 출력됩니다.)

4. Retype new password: 새로운 비밀번호를 동일하게 다시 입력합니다.

5. passwd: all authentication tokens updated successfully. 메시지가 출력되며 비밀번호 변경이 완료됩니다.

한양생명과학기술원
HY Institute of Bioscience and Biotechnology

4. SLURM 활용

4. SLURM 활용 (1. 주요 명령어)

※ 마스터 서버에 접속 한 후 계산 노드를 사용하기 위해서는 자원관리자(Resource Manager)를 사용해야 합니다.
IBB 서버에는 SLURM(Simple LinUx Resource Manager) 이라는 자원관리자가 사용되고 있습니다.

다음은 서버의 자원 상태를 확인할 때 사용할 수 있는 slurm 명령어입니다.

1. <sinfo> 또는 <sinfolong> 명령어를 통하여 각 노드 정보를 조회할 수 있습니다.

```
(base) [testuser2@ibb-master ~]$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
cpu*         up 7-00:00:00      4   idle node[01-04]
a100         up 7-00:00:00      1   idle gpu01
rtx3090      up 7-00:00:00      1    mix gpu02
rtx3090      up 7-00:00:00      1   idle gpu03
(base) [testuser2@ibb-master ~]$ sinfolong
PARTITION        CPUS(A/I/O/T)  STATE  NODELIST       NODES GRES            TIMELIMIT   AVAIL_FEATURES
cpu*             0/608/0/608    idle   node[01-04]    4     (null)          7-00:00:00  (null)
a100             0/80/0/80      idle   gpu01          1     gpu:a100:8      7-00:00:00  (null)
rtx3090          20/60/0/80     mix    gpu02          1     gpu:rtx3090:4   7-00:00:00  (null)
rtx3090          0/80/0/80      idle   gpu03          1     gpu:rtx3090:4   7-00:00:00  (null)
```

-CPUS(A/I/O/T)
: 사용 가능한 CPU 자원의 개수를 확인할 수 있습니다.
(' A ' 는 사용중인 thread 개수 / ' I ' 는 사용 가능한 thread 개수 / ' T ' 는 전체 thread 개수를 나타냅니다.)

-STATE
: 사용되지 않고 있는 노드는 'idle', 사용중인 노드는 'mix' 로 출력됩니다.

한양생명과학기술원
HY Institute of Bioscience and Biotechnology

2. <squeue> 또는 <squeuelong> 명령어를 통하여 제출된 작업 목록 및 정보를 조회할 수 있습니다.

```
(base) [testuser2@ibb-master ~]$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
              1094       cpu sleep60. testuser PD       0:00      1 (QOSMaxCpuPerUserLimit)
              1093       cpu sleep60. testuser  R       0:03      1 node02
              1092       cpu sleep60. testuser  R       0:06      1 node01
(base) [testuser2@ibb-master ~]$
(base) [testuser2@ibb-master ~]$ squeuelong
JOBID   NAME         STATE     USER           GROUP     PARTITION  QOS      NODE NODELIST(REASON)     CPUS TRES_PER_NODE        TIME_LIMIT  TIME_LEFT   PRIORITY
1094    sleep60.sh   PENDING   testuser2      testuser cpu        cpu-40   1    (QOSMaxCpuPerUserLim 20   N/A                  7-00:00:00  7-00:00:00  0.00000000023283
1093    sleep60.sh   RUNNING   testuser2      testuser cpu        cpu-40   1    node02               20   N/A                  7-00:00:00  6-23:59:54  0.00000000023283
1092    sleep60.sh   RUNNING   testuser2      testuser cpu        cpu-40   1    node01               20   N/A                  7-00:00:00  6-23:59:51  0.00000000023283
(base) [testuser2@ibb-master ~]$
```

- 작업 ID, 작업 이름, 작업 상태, 사용자 아이디, 사용중인 CPU thread 수 등의 정보를 확인할 수 있습니다.
- 제출된 작업이 돌아가지 않을 경우 'STATE' 항목에 'PENDING' 으로 표시되며,
  'NODELIST(REASON)' 항목에서 그 이유를 확인할 수 있습니다.

```
(base) [testuser2@ibb-master ~]$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
(base) [testuser2@ibb-master ~]$
```

- 실행중인 작업이 없을 때에는 위와 같이 항목 이름이 적힌 줄만 출력됩니다.

한양생명과학기술원
HY Institute of Bioscience and Biotechnology

3. <scancel **[job ID]**> 명령어를 통하여 제출된 작업 수행을 취소할 수 있습니다.

```
(base) [testuser2@ibb-master ~]$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
              1096       cpu    sleep testuser  R       0:03      1 node02
              1095       cpu    sleep testuser  R       0:06      1 node01
(base) [testuser2@ibb-master ~]$
(base) [testuser2@ibb-master ~]$ scancel 1096
(base) [testuser2@ibb-master ~]$
(base) [testuser2@ibb-master ~]$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
              1095       cpu    sleep testuser  R       0:29      1 node01
(base) [testuser2@ibb-master ~]$
```

4. <scontrol show node **[node name]**> 명령어를 통하여 해당 노드의 자세한 상태를 조회할 수 있습니다.

```
(base) [testuser2@ibb-master ~]$ scontrol show node  node01
NodeName=node01 Arch=x86_64 CoresPerSocket=1
   CPUAlloc=0 CPUEfctv=152 CPUTot=152 CPULoad=0.00
   AvailableFeatures=(null)
   ActiveFeatures=(null)
   Gres=(null)
   NodeAddr=node01 NodeHostName=node01 Version=22.05.2
   OS=Linux 4.18.0-425.3.1.el8.x86_64 #1 SMP Wed Nov 9 20:13:27 UTC 2022
   RealMemory=1 AllocMem=0 FreeMem=505274 Sockets=152 Boards=1
   State=IDLE ThreadsPerCore=1 TmpDisk=0 Weight=1 Owner=N/A MCS_label=N/A
   Partitions=cpu
   BootTime=2022-12-20T16:18:03 SlurmdStartTime=2023-03-15T13:28:04
   LastBusyTime=2023-06-16T17:59:03
   CfgTRES=cpu=152,mem=1M,billing=152
   AllocTRES=
   CapWatts=n/a
   CurrentWatts=0 AveWatts=0
   ExtSensorsJoules=n/s ExtSensorsWatts=0 ExtSensorsTemp=n/s
```

※ 각 노드의 이름은 다음과 같습니다.

- CPU : node01 / node02 / node03 / node04
- GPU a100 : gpu01
- GPU rtx3090 : gpu02 / gpu03

5. <scontrol show job **[job ID]**> 명령어를 통하여 제출된 작업의 자세한 상태를 조회할 수 있습니다.
   (**[job ID]**는 squeue 명령어를 통해 확인 가능)

```
(base) [testuser2@ibb-master ~]$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
              1097       cpu    sleep testuser  R       0:05      1 node01
(base) [testuser2@ibb-master ~]$
(base) [testuser2@ibb-master ~]$ scontrol show job 1097
JobId=1097 JobName=sleep
   UserId=testuser2(9107) GroupId=testuser2(9107) MCS_label=N/A
   Priority=1 Nice=0 Account=normal QOS=cpu-40 WCKey=*
   JobState=RUNNING Reason=None Dependency=(null)
   Requeue=1 Restarts=0 BatchFlag=1 Reboot=0 ExitCode=0:0
   RunTime=00:00:11 TimeLimit=00:10:00 TimeMin=N/A
   SubmitTime=2023-06-16T18:09:29 EligibleTime=2023-06-16T18:09:29
   AccrueTime=2023-06-16T18:09:29
   StartTime=2023-06-16T18:09:30 EndTime=2023-06-16T18:19:30 Deadline=N/A
   SuspendTime=None SecsPreSuspend=0 LastSchedEval=2023-06-16T18:09:30 Scheduler=Main
   Partition=cpu AllocNode:Sid=ibb-master:931707
   ReqNodeList=(null) ExcNodeList=(null)
   NodeList=node01
   BatchHost=node01
   NumNodes=1 NumCPUs=10 NumTasks=1 CPUs/Task=10 ReqB:S:C:T=0:0:*:*
   TRES=cpu=10,node=1,billing=10
   Socks/Node=* NtasksPerN:B:S:C=0:0:*:* CoreSpec=*
   MinCPUsNode=10 MinMemoryNode=0 MinTmpDiskNode=0
   Features=(null) DelayBoot=00:00:00
   OverSubscribe=OK Contiguous=0 Licenses=(null) Network=(null)
   Command=/lustre/home/testuser2/batch.sh
   WorkDir=/lustre/home/testuser2
   StdErr=/lustre/home/testuser2/sleep.1097
   StdIn=/dev/null
   StdOut=/lustre/home/testuser2/sleep.1097
   Power=


(base) [testuser2@ibb-master ~]$
```


한양생명과학기술원
HY Institute of Bioscience and Biotechnology

4. SLURM 활용 (2. srun 통해 자원 할당 받기 - CPU 자원)

다음은 <srun> 명령어를 통해 자원을 할당 받고 노드에 접속하는 방법입니다.

1. <srun --qos=cpu-40 --cpus-per-task=5 --pty bash> 와 같이 입력하여 CPU 노드에 접속 할 수 있습니다.
   (접속되면 아이디 뒷부분이 접속한 노드 이름(node01)으로 변경된 것을 확인할 수 있습니다.)

```
(base) [testuser2@ibb-master ~]$ srun --qos=cpu-40 --cpus-per-task=5 --pty bash
(base) [testuser2@node01 ~]$
(base) [testuser2@node01 ~]$ squeuelong
JOBID   NAME         STATE     USER          GROUP    PARTITION  QOS      NODE NODELIST(REASON)     CPUS TRES_PER_NODE
1042    bash         RUNNING   testuser2     testuser cpu        cpu-40   1    node01               5    N/A
(base) [testuser2@node01 ~]$
(base) [testuser2@node01 ~]$ exit
exit
(base) [testuser2@ibb-master ~]$
```

- --qos=cpu-40

: 사용자가 신청한 노드에 따라 qos 값이 설정되어 있습니다. CPU 40threads를 신청한 경우 위와 같은 qos값을 갖습니다.

- --cpus-per-task=5

: 실제 사용할 CPU thread 개수를 지정합니다. 이 옵션을 생략하면 1개의 thread를 사용하게 됩니다.

- --pty bash

: <srun> 명령어를 통해 현재 비어있는 노드에 접속하기 위해 사용되는 옵션입니다.

※ 노드에 접속해 있는 동안 작업을 실행하지 않아도 자원을 점유하게 됩니다.
때문에 작업을 완료하면 <exit> 명령어를 입력하여 노드 접속을 해제하고 불필요한 자원 사용을 방지합니다.

한양생명과학기술원

2. <srun --qos=rtx3090-8 --partition=rtx3090 --gres=gpu:rtx3090:1 --pty bash> 와 같이 입력하여 GPU 노드에 접속 할 수 있습니다.
(접속되면 아이디 뒷부분이 접속한 노드 이름(gpu02)으로 변경된 것을 확인할 수 있습니다.)

```
(base) [testuser2@ibb-master ~]$ srun --qos=rtx3090-8 --partition=rtx3090 --gres=gpu:rtx3090:1 --pty bash
(base) [testuser2@gpu02 ~]$
(base) [testuser2@gpu02 ~]$ squeuelong
JOBID   NAME         STATE     USER          GROUP     PARTITION  QOS      NODE NODELIST(REASON)     CPUS TRES_PER_NODE
1043    bash         RUNNING   testuser2     testuser rtx3090    rtx3090- 1    gpu02                20   gres:gpu:rtx3090:1
(base) [testuser2@gpu02 ~]$
(base) [testuser2@gpu02 ~]$ exit
exit
(base) [testuser2@ibb-master ~]$
```

- --qos=rtx3090-8

: 사용자가 신청한 노드에 따라 qos 값이 설정되어 있습니다. GPU RTX3090 8장을 신청한 경우 위와 같은 qos값을 갖습니다.

- --partition=rtx3090

: GPU 자원을 사용하기 위해서는 partition 옵션을 추가해주어야 합니다. a100, rtx3090 중 적절한 값을 입력합니다.

- --gres=gpu:rtx3090:1

: 사용할 GPU 종류와 개수를 지정합니다. rtx3090 1장을 사용할 경우 위와 같이 입력합니다.

- GPU 자원의 경우 a100은 1장당 기본 10개, rtx3090은 1장당 기본 20개의 CPU threads가 할당되도록 설정되어 있습니다.

※ 노드에 접속해 있는 동안 작업을 실행하지 않아도 자원을 점유하게 됩니다.
때문에 작업을 완료하면 <exit> 명령어를 입력하여 노드 접속을 해제하고 불필요한 자원 사용을 방지합니다.

한양생명과학기술원

4. SLURM 활용 (3. sbatch 통해 작업 제출하기 - CPU 자원)

다음은 <sbatch> 명령어를 통하여 자원을 할당 받고 작업을 제출하는 방법입니다.

1. <sbatch --qos=cpu-40 --cpus-per-task=5 --time=0-90:00:00 [실행파일]> 와 같이 입력하여 CPU 노드에 작업을 제출합니다.
   (작업 제출이 완료되면 Submitted batch job 1104 와 같이 작업 ID가 출력됩니다.)

```
(base) [testuser2@ibb-master ~]$ sbatch --qos=cpu-40 --cpus-per-task=5 --time=0-90:00:00 sleep60.sh
Submitted batch job 1104
(base) [testuser2@ibb-master ~]$
(base) [testuser2@ibb-master ~]$ squeuelong
JOBID   NAME         STATE     USER          GROUP    PARTITION  QOS      NODE NODELIST(REASON)     CPUS TRES_PER_NODE        TIME_LIMIT  TIME_LEFT
1104    sleep60.sh   RUNNING   testuser2     testuser cpu        cpu-40   1    node01               5    N/A                  3-18:00:00  3-17:59:54
(base) [testuser2@ibb-master ~]$
```

- --qos=cpu-40

: 사용자가 신청한 노드에 따라 qos 값이 설정되어 있습니다. CPU 40threads를 신청한 경우 위와 같은 qos값을 갖습니다.

- --cpus-per-task=5

: 할당 받을 CPU thread 개수를 지정합니다. 이 옵션을 생략하면 1개의 thread를 사용하게 됩니다.

- --time=0-90:00:00 또는 --time=60

: 작업 제출시 예상되는 최대 시간을 지정합니다. (day-hh:mm:ss 또는 minutes)

- <sbatch> 명령어는 표준 출력을 output 파일로 저장합니다.

한양생명과학기술원

4. SLURM 활용 (3. sbatch 통해 작업 제출하기 - GPU 자원)

2. <sbatch --qos=a100-8 --partition=a100 --gres=gpu:a100:2 --time=60 [실행파일]> 와 같이 입력하여 GPU 노드에 작업을 제출합니다.
(작업 제출이 완료되면 Submitted batch job 1106 와 같이 작업 ID가 출력됩니다.)

```
(base) [testuser2@ibb-master ~]$ sbatch --qos=a100-8 --partition=a100 --gres=gpu:a100:2 --time=60 sleep60.sh
Submitted batch job 1106
(base) [testuser2@ibb-master ~]$
(base) [testuser2@ibb-master ~]$ squeuelong
JOBID   NAME         STATE    USER       GROUP    PARTITION  QOS      NODE NODELIST(REASON)    CPUS TRES_PER_NODE       TIME_LIMIT  TIME_LEFT
1106    sleep60.sh   RUNNING  testuser2  testuser a100       a100-8   1    gpu01               20   gres:gpu:a100:2     1:00:00     59:58
(base) [testuser2@ibb-master ~]$
```

- --qos=a100-8
: 사용자가 신청한 노드에 따라 qos 값이 설정되어 있습니다. GPU A100 8장을 신청한 경우 위와 같은 qos값을 갖습니다.

- --partition=a100
: GPU 자원을 사용하기 위해서는 partition 옵션을 추가해주어야 합니다. a100, rtx3090 중 적절한 값을 입력합니다.

- --gres=gpu:a100:2
: 사용할 GPU 종류와 개수를 지정합니다. a100 2장을 사용할 경우 위와 같이 입력합니다.

- --time=0-90:00:00 또는 --time=60
: 작업 제출시 예상되는 최대 시간을 지정합니다. (day-hh:mm:ss 또는 minutes)

- <sbatch> 명령어는 표준 출력을 output 파일로 저장합니다.

한양생명과학기술원

4. SLURM 활용 (4. 주의사항)

- SLURM을 이용하지 않고 마스터 노드에서 작업을 수행하는 것은 전체 서버에 영향을 미치므로 피하도록 합니다.
- srun 은 작업이 완료되어도 exit 로 나가지 않으면 계속 노드 접근 상태가 유지되므로 대기시간을 줄이기 위해 간단한 디버깅 용도로만 짧게 사용하고 가능하면 sbatch 를 사용하는 것을 권장합니다.
- `sbatch` 작업 제출시 예상되는 최대 시간을 지정하도록 합니다.
- 활성화된 `slurm` 세션을 이용하여 `ssh` 등으로 노드에 직접 접근하는 것은 다른 사용자와의 충돌을 야기할 수 있으므로 금하도록 합니다.
- qos 값은 사용자가 신청한 노드 단위에 따라 다음과 같은 형식으로 설정되어 있습니다.

- CPU 10 threads : cpu-10 / CPU 20 threads : cpu-20 / CPU 30 threads : cpu-30 ...
- GPU A100 1장 : a100-1 / GPU A100 2장 : a100-2 / GPU A100 3장 : a100-3 ...
- GPU RTX3090 1장 : rtx3090-1 / GPU RTX3090 2장 : rtx3090-2 / GPU RTX3090 3장 : rtx3090-3 ...

- .sh 파일의 경우 첫 줄에 ' #!/bin/bash ' 를 입력해주어야 정상적으로 파일내 구문 실행이 가능합니다.

한양생명과학기술원
HY Institute of Bioscience and Biotechnology

※ sbatch 의 경우 자주 사용하는 옵션들을 쉘 스크립트 파일 내에 넣어주면 매번 반복하여 입력하지 않고 실행이 가능합니다.

```
#!/bin/bash

#SBATCH --job-name=sleep                    # Job name
#SBATCH --output=sleep.%j                   # Name of stdout output file (%j expands to %jobId)
#SBATCH --qos=cpu-40                        # QoS
#SBATCH --partition=cpu                     # queue or partiton name
#SBATCH --time=0-00:10:00                   # Max Run time (day-hh:mm:ss)
#SBATCH --cpus-per-task=10                  # number of cpus required per task


date
sleep 60
date
```

- 일반적인 스크립트 파일에서 # 은 주석으로 처리 되지만,
  SLURM 에서는 맨 앞에 있는 # 1개 와 sbatch 구문이 문법으로 처리됩니다.

```
(base) [testuser2@ibb-master ~]$ sbatch batch.sh
Submitted batch job 1068
(base) [testuser2@ibb-master ~]$
(base) [testuser2@ibb-master ~]$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
              1068       cpu    sleep testuser  R       0:04      1 node01
(base) [testuser2@ibb-master ~]$
```

- 위와 같이 sbatch 구문이 포함된 쉘 스크립트 파일은 바로 작업 제출이 가능합니다.

한양생명과학기술원
HY Institute of Bioscience and Biotechnology

5. Environment-module

한양생명과학기술원
HY Institute of Bioscience and Biotechnology

5. Environment-module

Environment-module 은 환경변수를 module 화 하여 사전에 관리자를 통해서
설치된 패키지에 대한 환경변수를 손쉽게 load 하거나 swap 할 수 있는 도구 입니다.

1. <module avail> 또는 <module av> 명령어로 load 할 수 있는 모듈 목록을 조회할 수 있습니다.

```
(base) [testuser2@ibb-master ~]$ module avail

------------------------------------------ /opt/ohpc/pub/modulefiles ------------------------------------------
   AmpliconNoise/1.29      breakdancer/1.4.5          infernal/1.1.4          randfold
   CAP3                    bwa/0.7.15                 intel/2022.2.1          rmblast/2.13.0
   Cytoscape/3.9.1         bwa/0.7.17          (D)    java/11.0.17            rsem/1.3.3
   ETCHING/1.4.2           cdhit/4.8.1                java/17.0.6      (D)    saintexpress/3.6.3
   EasyBuild/4.6.2         cdk/5.0                    jellyfish/2.2.7         salmon/0.9.1          (D)
   HiC-Pro/3.1.0           charliecloud/0.15          libfabric/1.13.0        salmon/1.4.0
   IGVTools/2.16.0         circos/0.69-9              lobSTR/4.0.6            salmon/1.8.0
                                                                              
   bowtie/2.3.4.1    (D)   hwloc/2.7.0                python/3.11.3    (D)    velvet/1.2.10

  Where:
   D:  Default Module
   L:  Module is loaded

If the avail list is too long consider trying:

"module --default avail" or "ml -d av" to just list the default modules.
"module overview" or "ml ov" to display the number of modules for each name.

Use "module spider" to find all possible modules and extensions.
Use "module keyword key1 key2 ..." to search for all possible modules matching any of the "keys".


(base) [testuser2@ibb-master ~]$
```

※ 서버에 설치된 패키지 목록은 IBB 홈페이지(http://ibb.hanyang.ac.kr/) 커뮤니티 자료실에서도 확인 가능합니다.

한양생명과학기술원
HY Institute of Bioscience and Biotechnology

5. Environment-module

```
(base) [testuser2@ibb-master ~]$ module list

Currently Loaded Modules:
  1) cuda/11.8   2) singularity/3.7.1   3) ohpc


(base) [testuser2@ibb-master ~]$ module load bowtie
(base) [testuser2@ibb-master ~]$ module list

Currently Loaded Modules:
  1) cuda/11.8   2) singularity/3.7.1   3) ohpc   4) bowtie/2.3.4.1


(base) [testuser2@ibb-master ~]$ module swap cuda/11.8 cuda/12.0

The following have been reloaded with a version change:
  1) cuda/11.8 => cuda/12.0

(base) [testuser2@ibb-master ~]$ module list

Currently Loaded Modules:
  1) singularity/3.7.1   2) ohpc   3) bowtie/2.3.4.1   4) cuda/12.0


(base) [testuser2@ibb-master ~]$ module unload bowtie
(base) [testuser2@ibb-master ~]$ module list

Currently Loaded Modules:
  1) singularity/3.7.1   2) ohpc   3) cuda/12.0


(base) [testuser2@ibb-master ~]$ module purge
(base) [testuser2@ibb-master ~]$ module list
No modules loaded
(base) [testuser2@ibb-master ~]$
```

2. <module list> 명령어로
   현재 load 되어있는 모듈 목록을 조회할 수 있습니다.

3. <module load **[모듈명]**> 명령어로 특정 모듈을 load 합니다.

4. <module swap **[모듈명1] [모듈명2]**> 명령어로
   특정 모듈을 다른 모듈로 교체합니다.

5. <module unload **[모듈명]**> 명령어로 특정 모듈을 unload 합니다.

6. <module purge> 명령어로
   현재 load 되어있는 모든 모듈을 unload 합니다.

※ p.36 처럼 쉘 스크립트 파일 내에 명령어를 추가하여 사용하거나
자신의 ~/.bashrc 파일 내에 명령어를 추가하여
로그인시 기본 실행 환경으로 구성 가능합니다.

한양생명과학기술원

6. Anaconda

한양생명과학기술원
HY Institute of Bioscience and Biotechnology

6. Anaconda (1. 주요 명령어)

Anaconda 는 여러가지 패키지를 관리자 권한없이 설치 하고 프로젝트 별로 다른 환경을 구성 할 수 있는 배포도구 입니다.
모듈에 없는 패키지를 설치하거나 필요한 패키지만 설치하여 의존성(Dependency) 문제를 해결하는데 활용할 수 있습니다.

IBB 서버에는 기본적으로 Anaconda가 설치되어 있으며 로그인시 자동으로 활성화됩니다.
(Anaconda 가 활성화 된 상태에서는 userID 좌측에 (base) 표시가 출력됩니다.)

```
(base) [testuser2@ibb-master ~]$
(base) [testuser2@ibb-master ~]$ conda env list
# conda environments:
#
base                  *  /opt/ohpc/pub/apps/anaconda3

(base) [testuser2@ibb-master ~]$
(base) [testuser2@ibb-master ~]$
```

1. <conda env list> 명령어를 통하여 설치되어 있는 가상환경 리스트를 조회할 수 있습니다.
   (활성화 되어있는 가상환경 이름 옆에 * 표시가 출력됩니다.)

한양생명과학기술원

```
(base) [testuser2@ibb-master ~]$ conda create -n bio python=3.7
Retrieving notices: ...working... done
Collecting package metadata (current_repodata.json): done
Solving environment: failed with repodata from current_repodata.json, will retry with next repodata source.
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /home/testuser2/.conda/envs/bio

  added / updated specs:
    - python=3.7


The following packages will be downloaded:
```

```
(base) [testuser2@ibb-master ~]$
(base) [testuser2@ibb-master ~]$ conda env list
# conda environments:
#
bio                      /home/testuser2/.conda/envs/bio
base                  *  /opt/ohpc/pub/apps/anaconda3

(base) [testuser2@ibb-master ~]$
(base) [testuser2@ibb-master ~]$
```

2. <conda create -n **[가상환경명]** python=**[python 버전]**> 와 같이 입력하여 가상환경을 생성 할 수 있습니다.

   (가상환경 리스트에 'bio' 라는 가상환경이 추가된 것을 확인할 수 있습니다.)

   (python 옵션을 생략하면 최신 버전의 python을 설치합니다.)

한양생명과학기술원

```
(base) [testuser2@ibb-master ~]$
(base) [testuser2@ibb-master ~]$ conda activate bio
(bio) [testuser2@ibb-master ~]$
(bio) [testuser2@ibb-master ~]$ conda deactivate
(base) [testuser2@ibb-master ~]$
(base) [testuser2@ibb-master ~]$ conda deactivate
[testuser2@ibb-master ~]$
[testuser2@ibb-master ~]$
```

```
(base) [testuser2@ibb-master ~]$
(base) [testuser2@ibb-master ~]$ conda remove -n bio --all

Remove all packages in environment /home/testuser2/.conda/envs/bio:


## Package Plan ##

  environment location: /home/testuser2/.conda/envs/bio

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
(base) [testuser2@ibb-master ~]$
(base) [testuser2@ibb-master ~]$ conda env list
# conda environments:
#
base                  *  /opt/ohpc/pub/apps/anaconda3

(base) [testuser2@ibb-master ~]$
```

3. <conda activate **[가상환경명]**>
   명령어를 통하여 특정 가상환경을 활성화 할 수 있습니다.
   (userID 좌측에 출력되는 가상환경명 이 변경됩니다.)

4. <conda deactivate>
   명령어를 통하여 가상환경을 비활성화 할 수 있습니다.
   (base 상태에서 명령어를 실행하면 Anaconda를 종료합니다.)

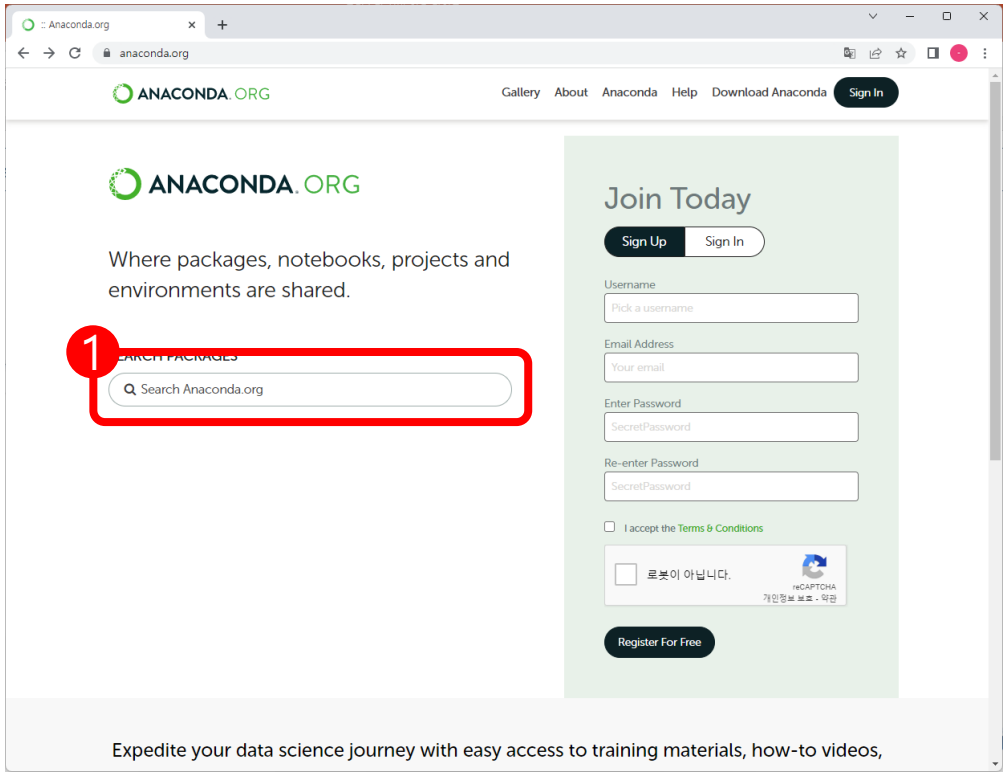
5. <conda remove -n **[가상환경명]** --all> 와 같이 입력하여
   특정 가상환경을 제거할 수 있습니다. (가상환경 리스트에서
   'bio' 라는 가상환경이 제거된 것을 확인할 수 있습니다.)

※ p.36 처럼 쉘 스크립트 파일 내에 추가하여 사용하거나
   자신의 ~/.bashrc 파일 내에 명령어를 추가하여
   로그인시 기본 실행 환경으로 구성 가능합니다.

한양생명과학기술원

6. Anaconda (2. 패키지 설치)

Anaconda 를 통해 설치 할 수 있는 패키지는 'https://anaconda.org' 에서 검색할 수 있습니다.

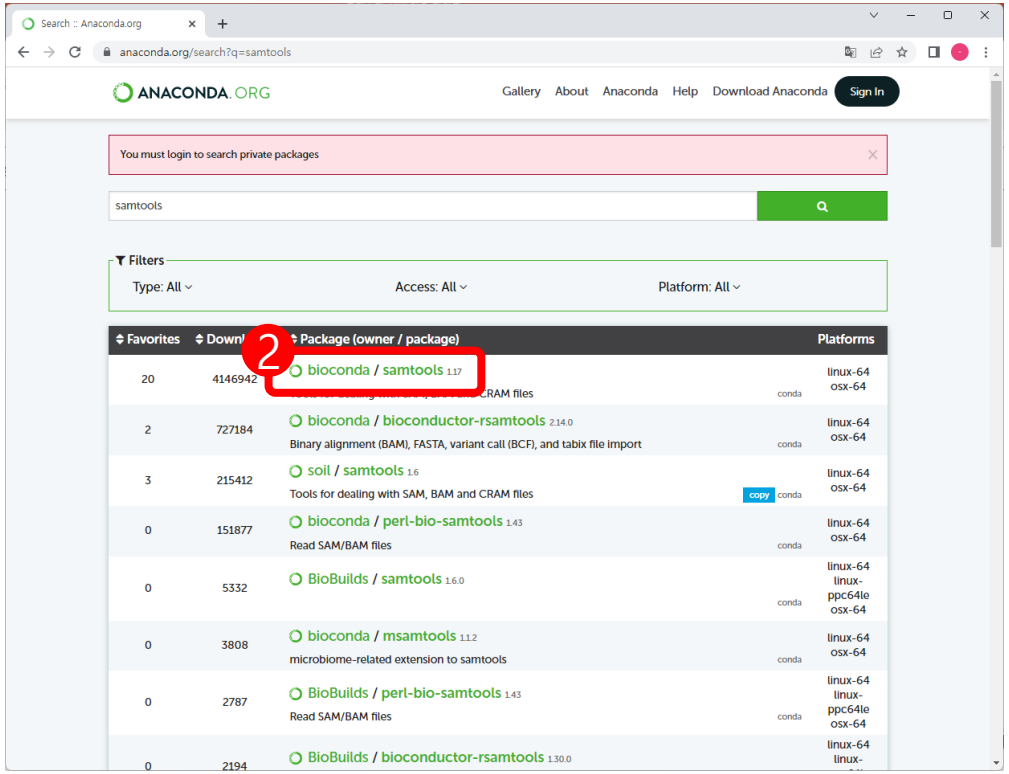


다음은 samtools 라는 패키지를 Anaconda 를 통해 설치하는 과정입니다.

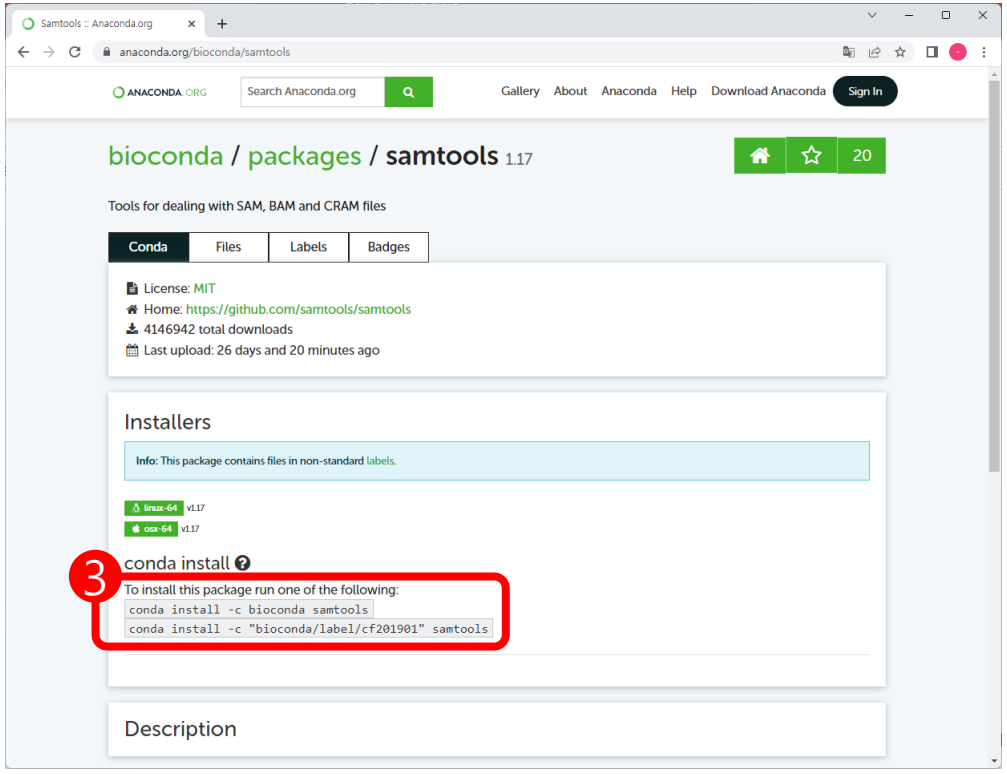
1. https://anaconda.org 에서 설치하고자 하는 패키지를 검색합니다.
(Google 에 'conda install 패키지명' 으로 검색하여도 접속 가능합니다.)

한양생명과학기술원
HY Institute of Bioscience and Biotechnology

6. Anaconda (2. 패키지 설치)



2. 검색 된 목록 중에서 원하는 패키지를 찾아 선택합니다.



3. conda install 명령어를 복사합니다.

한양생명과학기술원

6. Anaconda (2. 패키지 설치)

```
(base) [testuser2@ibb-master ~]$
(base) [testuser2@ibb-master ~]$ conda activate bio
(bio) [testuser2@ibb-master ~]$
(bio) [testuser2@ibb-master ~]$ conda install -c bioconda samtools
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /home/testuser2/.conda/envs/bio

  added / updated specs:
    - samtools


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    c-ares-1.19.0              |       h5eee18b_0         118 KB
    curl-8.1.1                 |       h5eee18b_0          80 KB
    krb5-1.19.4                |       h568e23c_0         1.3 MB

Downloading and Extracting Packages

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
(bio) [testuser2@ibb-master ~]$
```

4. 패키지를 설치하고자 하는 가상환경을 활성화 한 상태에서
   복사한 명령어를 서버에 입력하여 설치를 진행합니다.

```
(bio) [testuser2@ibb-master ~]$
(bio) [testuser2@ibb-master ~]$ conda list
# packages in environment at /home/testuser2/.conda/envs/bio:
#
# Name                    Version                   Build  Channel
_libgcc_mutex             0.1                       main
_openmp_mutex             5.1                       1_gnu
bzip2                     1.0.8                h7b6447c_0
c-ares                    1.19.0               h5eee18b_0
ca-certificates           2023.05.30           h06a4308_0
certifi                   2022.12.7         py37h06a4308_0
curl                      8.1.1                h5eee18b_0
krb5                      1.19.4               h568e23c_0
ncurses                   6.4                  h6a678d5_0
openssl                   1.1.1u               h7f8727e_0
pip                       22.3.1            py37h06a4308_0
python                    3.7.16               h7a1cb2a_0
readline                  8.2                  h5eee18b_0
samtools                  1.5                           1    bioconda
setuptools                65.6.3            py37h06a4308_0
sqlite                    3.41.2               h5eee18b_0
tk                        8.6.12               h1ccaba5_0
wheel                     0.38.4            py37h06a4308_0
xz                        5.4.2                h5eee18b_0
zlib                      1.2.13               h5eee18b_0
(bio) [testuser2@ibb-master ~]$
(bio) [testuser2@ibb-master ~]$
```

5. <conda list> 명령어를 통하여
   현재 가상환경에 설치된 패키지 목록을 조회할 수 있습니다.

한양생명과학기술원

7. 한양 VPN(Virtual Private Network)

7. 한양 VPN(Virtual Private Network)

- 외부에서의 접근하거나 장소 이동이 잦아 특정 IP 내에서만 서버에 접속 하는 것이 어려울 경우 한양 VPN 서비스 사용 권장합니다.
- 한양대학교 포털에서 신청 후 학교 정보인프라팀의 승인을 거쳐 사용 가능합니다.
(당일 승인 불가, 한학기 단위로 신청 및 연장)
- 보안 및 라이센스 문제로 최대 연결 시간이 4시간으로 고정되어 있습니다.
- 한양 VPN 프로그램을 활성화 한 경우 정상적으로 서버 접근이 가능합니다.
- 한양 VPN 서비스 안내 페이지 : http://vpninfo.hanyang.ac.kr/

문의처

한양생명과학기술원
바이오공동기기센터 BI/컴퓨팅 코어 담당자

Tel. 02-2220-1742
E-mail. ibbbicomp@hanyang.ac.kr

한양생명과학기술원 http://ibb.hanyang.ac.kr/