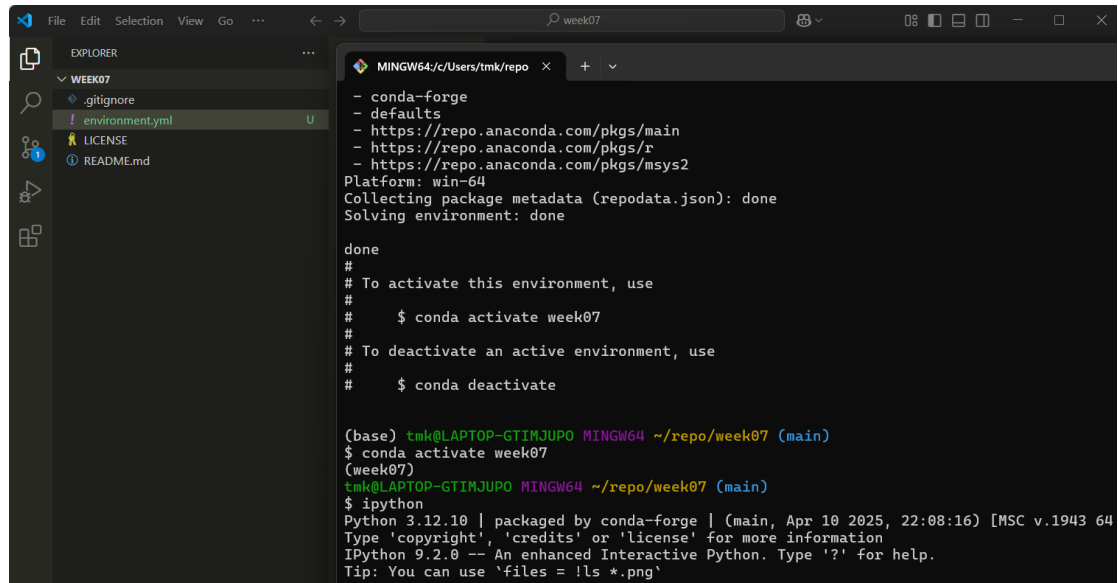


第七周学习报告

安装完成并激活 jupyterlab



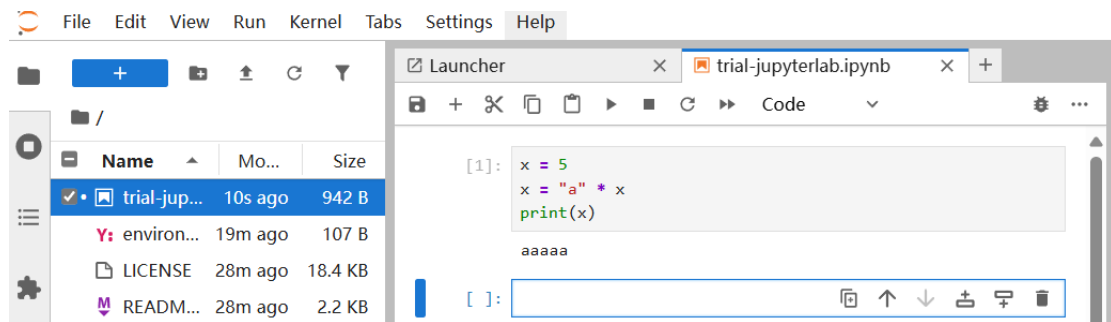
```
MINGW64/c/Users/tmk/repo x +
- conda-forge
- defaults
- https://repo.anaconda.com/pkgs/main
- https://repo.anaconda.com/pkgs/r
- https://repo.anaconda.com/pkgs/msys2
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done

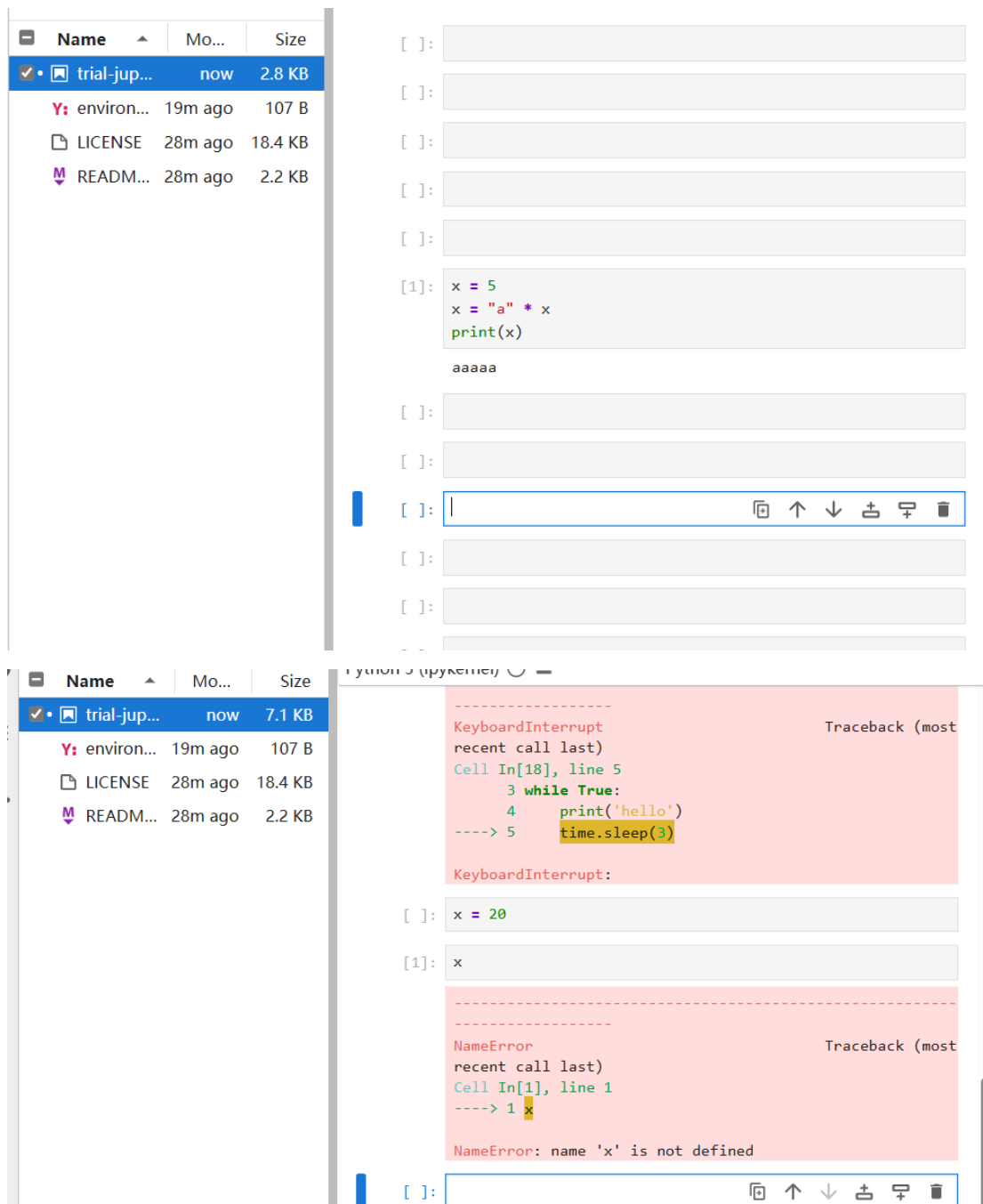
done
#
# To activate this environment, use
#
#   $ conda activate week07
#
# To deactivate an active environment, use
#
#   $ conda deactivate

(base) tmk@LAPTOP-GTIMJUPO MINGW64 ~/repo/week07 (main)
$ conda activate week07
(week07)
tmk@LAPTOP-GTIMJUPO MINGW64 ~/repo/week07 (main)
$ ipython
Python 3.12.10 | packaged by conda-forge | (main, Apr 10 2025, 22:08:16) [MSC v.1943 64]
Type 'copyright', 'credits' or 'license' for more information
IPython 9.2.0 -- An enhanced Interactive Python. Type '?' for help.
Tip: You can use 'files = !ls *.png'
```

应用

在单元格 (Cell) 的命令模式下, 按 j 选择下一个, 按 k 选择上一个, 按 a 在上方添加, 按 b 在下方添加, 按 dd 删除, 按住 Shift 多选, 按 x 剪切, 按 c 复制, 按 v 粘贴, 按 Shift+M 合并, 按 z 撤销, 按 Shift+Z 重做, 按 Shift+L 显示/隐藏代码行号





Parquet 是一种开源的列式存储文件格式，专为高效处理和存储大规模数据而设计，广泛应用于大数据和分析领域。下面为你详细介绍其特点：

- 高效的压缩和编码**：**Parquet 支持多种压缩算法（如 Snappy、Gzip、LZO 等）和编码技术（如 Run Length Encoding、Dictionary Encoding 等）。这些技术可以显著减少数据的存储空间，从而降低存储成本，同时还能减少数据传输量，提高数据处理的效率。
- 列式存储**：**列式存储使得在处理只涉及部分列的查询时，无需读取整个行数据，仅读取需要的列，从而减少了 I/O 开销，提高了查询性能。例如，在数据分析中，若只需要分析某几列的数据，列式存储可以快速定位并读取这些数据，而无需读取其他无关列。
- 支持复杂数据类型**：**Parquet 能够处理如嵌套结构、数组、Map 等复杂的数据类型，这使得它非常适合存储半结构化和结构化的数据，例如 JSON 或 XML 数据。
- 可扩展性**：**Parquet 的设计具有良好的可扩展性，可以方便地添加新的压缩算法、编

码方式和元数据信息，以适应不同的应用场景和数据处理需求。

5. ****跨平台和跨语言支持****: Parquet 是一种独立于平台和语言的文件格式，几乎所有的大数据处理框架（如 Apache Hadoop、Spark、Presto 等）都提供了对 Parquet 的支持，方便不同系统之间的数据交换和共享。

6. ****数据分区****: Parquet 支持数据分区，可以根据数据的某个或多个列进行分区存储。在查询时，可以根据分区信息快速定位到需要的数据文件，进一步提高查询效率。

缺点

1. ****不适合实时写入****: 由于 Parquet 是为批量数据处理而设计的，在实时写入场景下，频繁的小数据写入会导致大量的小文件产生，增加元数据管理的负担，同时也会影响查询性能。

2. ****高并发写入性能较差****: 在高并发写入场景下，Parquet 的写入性能可能会受到影响，因为多个写入任务可能会相互竞争资源，导致写入效率下降。

3. ****不适合行级别的随机访问****: Parquet 的列式存储结构使得它在行级别的随机访问方面表现不佳，因为要访问某一行的数据，需要读取多个列的数据块，增加了 I/O 开销。