·····CPU·····

·····Slurm·····CPU·····

· Slurm ····· allocation·batch·extern · srun ····
····· MaxRSS ·····

·····

····· sample-steps.tsv····· ID ····· JobID.step··· S01·12345.0··· sacct ···
····· step·····

```
set -euo pipefail
: "${JOBS:?·····ID·····}" "${OUT:?·····}"
[[ "$JOBS" =~ ^[0-9]+(,[0-9]+)*$ ]] || exit 2
test ! -e "$OUT" && test ! -L "$OUT" || exit 2
out="$OUT"
mkdir "$out"
sacct --version > "$out/version.txt"
sacct -j "$JOBS" --parsable2 --units=K \
  --format=JobID%50,State%30,ExitCode,AllocCPUS,ElapsedRaw,TotalCPU,MaxRSS,NTasks \
  > "$out/steps.psv"
test -s "$out/steps.psv"
printf 'Saved: %s\n' "$out"
```

ElapsedRaw ·····TotalCPU ····· CPU ···AllocCPUS ···· CPU ···· TotalCPU / (ElapsedRaw × A
llocCPUS) ·····MaxRSS ·····

·· | ·· | ····

ElapsedRaw | ····· | ··CPU···

TotalCPU | ·····CPU·· | AllocCPUS·····

MaxRSS | ····· | ·····

State / ExitCode | ····· | ··State·····

·····

····· summarize_sacct.py·· sample-steps.tsv ····· OUT ·····
····· COMPLETED····· 0:0····· eligible·····

```
import csv, json, math, re
from pathlib import Path

import os
root = Path(os.environ.get('OUT', 'accounting-001'))
with open('sample-steps.tsv', newline='') as f:
    plan = list(csv.reader(f, delimiter='\t'))
if not plan or any(len(r) != 2 or not all(r) for r in plan):
    raise SystemExit('·····')
if len({r[0] for r in plan}) != len(plan) or len({r[1] for r in plan}) != len(plan):
    raise SystemExit('··ID·step·····')

def missing(text):
    return not text or text.strip().upper() in {'N/A', 'UNKNOWN', 'UNLIMITED'}

def cpu_seconds(text):
    if missing(text): return None
    days, clock = text.split('-', 1) if '-' in text else ('0', text)
    if not days.isdigit(): return None
    parts = clock.split(':')
    try: nums = [float(x) for x in parts]
    except ValueError: return None
    if not 1 <= len(nums) <= 3 or not all(math.isfinite(x) and x >= 0 for x in nums): return None
    if len(nums) >= 2 and nums[-1] >= 60: return None
    if len(nums) == 3 and nums[-2] >= 60: return None
    return int(days)*86400 + sum(x*60**i for i,x in enumerate(reversed(nums)))

def rss_kib(text):
    if missing(text): return None
    match = re.fullmatch(r'([0-9]+(?:[.][0-9]+)?)([KMGTPE]?)', text.strip().upper())
    if not match: return None
    scale = {'':1,'K':1,'M':1024,'G':1024**2,'T':1024**3,'P':1024**4,'E':1024**5}
    return float(match[1]) * scale[match[2]]

with open(root / 'steps.psv', newline='') as f:
    reader = csv.DictReader(f, delimiter='|')
    reader.fieldnames = [x.strip() for x in (reader.fieldnames or [])]
    records = {}
    for raw in reader:
        row = {k.strip(): (v or '').strip() for k,v in raw.items() if k is not None}
        if row.get('JobID'): records[row['JobID']] = row
report = []
for sample, step in plan:
    row = records.get(step)
    item = {'sample':sample, 'step':step, 'eligible':False}
    if row is None:
        item['reason'] = 'record_missing'
    else:
        rss = rss_kib(row.get('MaxRSS',''))
        cpu = cpu_seconds(row.get('TotalCPU',''))
        try: elapsed = int(row.get('ElapsedRaw',''))
        except ValueError: elapsed = None
        try: cpus = int(row.get('AllocCPUS',''))
        except ValueError: cpus = None
        ok = (row.get('State') == 'COMPLETED' and row.get('ExitCode') == '0:0'
              and row.get('NTasks') == '1' and rss is not None and rss > 0
              and elapsed is not None and elapsed > 0 and cpus is not None and cpus > 0
              and cpu is not None)
        item.update(state=row.get('State'), exit_code=row.get('ExitCode'),
                    maxrss_gib=None if rss is None else rss/1024**2,
                    elapsed_seconds=elapsed, total_cpu_seconds=cpu,
                    allocated_cpus=cpus, eligible=bool(ok))
        item['cpu_fraction'] = cpu/(elapsed*cpus) if ok else None
        if not ok: item['reason'] = 'failed_missing_or_incomparable'
    report.append(item)
with (root / 'sample-budget.json').open('x') as f:
    json.dump(report, f, indent=2)
print('planned',len(report),'eligible',sum(x['eligible'] for x in report))
```

·····

CPU fraction ·· 1 ····· CPU ·····I/O ·····
····· MaxRSS·TotalCPU · AllocCPUS ·····

·····

JobID.step ····· sbatch ····· batch step··· srun ····· ste
p···· sacct ····· step ····· ID····· step ·····
·····

·· reads ····· MaxRSS ··
·····
·····

·· MaxRSS ····· MaxRSS ·····
····· MaxRSS·····
·····

· MaxRSS ·····
····· sacct ·····
·····

····

SchedMD··sacct·····/····· https://slurm.schedmd.com/sacct.html

SchedMD··sbatch·····CPU·· https://slurm.schedmd.com/sbatch.html