# Millions — Data & Weekly Runbook (v1.0)

This doc captures how the **Millions** data flows, how to run weekly exports, and where each piece lives. It reflects the system state as of today (Week 1 pre‑season export is green).

## 1) Conceptual Map (why each file exists)

**Roadmap (canon)** → **Planner (working surface)** → **Exporter (HTML)**

* **Roadmap**: Authoritative schedule + market fields per game.
  + File(s): picks/millions/millions\_roadmap\_game.csv (season‑wide), picks/millions/millions\_weekly\_games.csv (subset convenience).
  + Holds: schedule (kickoff\_local, venue), home/away teams, lines (open/current/closing), Circa contest spreads (when available), rest days, DVOA per side (if you put it here), etc.
* **Planner**: Clean, per‑team rows, aligned to week/season, with **no placeholders**. The exporter reads this.
  + File: picks/millions/millions\_planner.csv.
  + Holds: team/opponent, home\_or\_away, DVOA projections (total/off/def), lines mapped to the team perspective, kickoff helper fields for sorting.
* **Exporter**: Builds the member dashboard HTML from the planner for a single week.
  + Script: scripts/millions\_export\_member\_html.py.
  + Output: picks/millions/exports/member\_week\_<W>.html.

Philosophy: keep the planner minimal and accurate. Enrich from real sources only. Strip any stale scaffolding.

## 2) Directory of Key Data Items

* picks/millions/millions\_roadmap\_game.csv
  + **Must‑have columns** (current schema):
    - Teams & schedule: week, game\_num, hometm/home\_team, vistm/away\_team, matchup, kickoff\_local, venue
    - Lines (spreads): open\_spread\_home/away, current\_spread\_home/away, closing\_spread\_home/away
    - Circa: circa\_spread\_home/away (may be empty until PDF ingest)
    - Rest: rest\_days\_home, rest\_days\_away, rest\_days\_diff
    - DVOA (optional but supported): total\_dvoa\_home/away, off\_dvoa\_home/away, def\_dvoa\_home/away, st\_dvoa\_home/away
    - Injury/Weather (optional): injuries\_key\_home/away, weather\_notes
  + **Planned additions**: open\_total, current\_total, closing\_total, circa\_total.
* picks/millions/millions\_planner.csv
  + **Required**: season, week, team, opponent, home\_or\_away.
  + **DVOA**: team\_total\_dvoa\_proj, opp\_total\_dvoa\_proj, dvoa\_diff\_proj (+ team\_off\_dvoa\_proj, team\_def\_dvoa\_proj, opp\_off\_dvoa\_proj, opp\_def\_dvoa\_proj).
  + **Lines**: current\_spread\_home/away, open\_spread\_home/away (game‑level, used to compute favorite), circa\_line (team‑oriented; may show TBD if not present).
  + **Kickoff helpers**: \_kickoff\_pt (string, PT), \_kickoff\_sort\_key (number; Thu→Mon order).
* DVOA Projections: data/dvoa/2025\_dvoa\_projections.csv (team‑level projections).
* **Audit output**: picks/millions/diagnostics/millions\_week\_audit.md (+ optional CSV flags).
* **Member HTML**: picks/millions/exports/member\_week\_<W>.html.

## 3) One‑Time Setup (already done)

* Planner rebuilt from roadmap (Week 1 only) with placeholders removed.
* \_kickoff\_pt + \_kickoff\_sort\_key added to planner; exporter sorts by this key (Thu→Fri→Sat→Sun→Mon last).
* DVOA shown as percentages; **Off/Def DVOA** mapped to team/opponent orientation; **Total DVOA** used for Top/Bottom 5 badges.
* Favorite (open/current) computed from home/away spread columns. If missing → displays **TBD**.

## 4) Pasteable Commands (Week Workflow)

Run from repo root: C:\Users\Spencer\OneDrive\Desktop\nfl25-agent

### A) Build/refresh the planner (single week)

python scripts/millions\_build\_planner.py ^  
 --season 2025 ^  
 --week 1 ^  
 --planner picks/millions/millions\_planner.csv ^  
 --schedule picks/millions/millions\_roadmap\_game.csv ^  
 --dvoa data/dvoa/2025\_dvoa\_projections.csv ^  
 --derive\_lines ^  
 --strip\_placeholders circa\_line result closing\_line line\_value pick\_side pick\_confidence notes

### B) Enrich planner with roadmap (lines + off/def DVOA + kickoff helpers)

python - <<'PY'  
import pandas as pd  
from pathlib import Path  
  
planner\_p = Path(r"picks/millions/millions\_planner.csv")  
roadmap\_p = Path(r"picks/millions/millions\_roadmap\_game.csv")  
  
p = pd.read\_csv(planner\_p)  
r = pd.read\_csv(roadmap\_p)  
  
# Normalize team codes  
for df in (p,):  
 for c in ("team","opponent","home\_or\_away"):  
 if c in df.columns:  
 df[c] = df[c].astype(str).str.strip().str.upper()  
for c in ("home\_team","away\_team","hometm","vistm"):  
 if c in r.columns:  
 r[c] = r[c].astype(str).str.strip().str.upper()  
  
# Orientation-agnostic keys  
p["\_key"] = p[["team","opponent"]].apply(lambda x: "::".join(sorted([x.team, x.opponent])), axis=1)  
if {"home\_team","away\_team"}.issubset(r.columns):  
 r["\_key"] = r[["home\_team","away\_team"]].apply(lambda x: "::".join(sorted([x.home\_team, x.away\_team])), axis=1)  
elif {"hometm","vistm"}.issubset(r.columns):  
 r["\_key"] = r[["hometm","vistm"]].apply(lambda x: "::".join(sorted([x.hometm, x.vistm])), axis=1)  
else:  
 raise SystemExit("Roadmap missing home/away team columns")  
  
keep = [c for c in (  
 "kickoff\_local","venue",  
 "open\_spread\_home","open\_spread\_away",  
 "current\_spread\_home","current\_spread\_away",  
 "closing\_spread\_home","closing\_spread\_away",  
 "circa\_spread\_home","circa\_spread\_away",  
 "rest\_days\_home","rest\_days\_away","rest\_days\_diff",  
 "off\_dvoa\_home","off\_dvoa\_away","def\_dvoa\_home","def\_dvoa\_away"  
) if c in r.columns]  
meta = r[["\_key"] + keep].drop\_duplicates("\_key")  
  
# Drop any old kickoff helper columns to avoid suffixing  
for col in ["kickoff\_pt","kickoff\_sort\_key","\_kickoff\_pt","\_kickoff\_sort\_key"]:  
 if col in p.columns:  
 del p[col]  
  
p = p.merge(meta, on="\_key", how="left")  
  
# Derive circa\_line by team orientation  
if 'home\_or\_away' in p.columns:  
 def pick\_circa(row):  
 if row.get('home\_or\_away') == 'HOME' and 'circa\_spread\_home' in p.columns:  
 return row.get('circa\_spread\_home')  
 if row.get('home\_or\_away') == 'AWAY' and 'circa\_spread\_away' in p.columns:  
 return row.get('circa\_spread\_away')  
 return row.get('circa\_line')  
 p['circa\_line'] = p.apply(pick\_circa, axis=1)  
  
# Map Off/Def DVOA to team/opponent perspective  
if {'off\_dvoa\_home','off\_dvoa\_away','def\_dvoa\_home','def\_dvoa\_away'}.issubset(p.columns):  
 def map\_off(row, side):  
 if row.get('home\_or\_away') == 'HOME':  
 return row.get('off\_dvoa\_home' if side=='team' else 'off\_dvoa\_away')  
 else:  
 return row.get('off\_dvoa\_away' if side=='team' else 'off\_dvoa\_home')  
 def map\_def(row, side):  
 if row.get('home\_or\_away') == 'HOME':  
 return row.get('def\_dvoa\_home' if side=='team' else 'def\_dvoa\_away')  
 else:  
 return row.get('def\_dvoa\_away' if side=='team' else 'def\_dvoa\_home')  
 p['team\_off\_dvoa\_proj'] = p.apply(lambda r: map\_off(r,'team'), axis=1)  
 p['opp\_off\_dvoa\_proj'] = p.apply(lambda r: map\_off(r,'opp'), axis=1)  
 p['team\_def\_dvoa\_proj'] = p.apply(lambda r: map\_def(r,'team'), axis=1)  
 p['opp\_def\_dvoa\_proj'] = p.apply(lambda r: map\_def(r,'opp'), axis=1)  
  
p.drop(columns=['\_key'], inplace=True, errors='ignore')  
p.to\_csv(planner\_p, index=False)  
print("Planner enriched:", planner\_p)  
PY

### C) Add kickoff PT + robust sort key from cleaned season schedule

Uses data/2025\_nfl\_schedule\_cleaned.csv columns: week, date, time, vistm, hometm.

python - <<'PY'  
import pandas as pd  
from pathlib import Path  
from datetime import datetime, timedelta  
  
PLANNER = Path(r"picks/millions/millions\_planner.csv")  
SCHED = Path(r"data/2025\_nfl\_schedule\_cleaned.csv")  
WEEK = 1  
  
p = pd.read\_csv(PLANNER)  
s = pd.read\_csv(SCHED)  
  
for c in ("team","opponent","home\_or\_away"):  
 if c in p.columns:  
 p[c] = p[c].astype(str).str.strip().str.upper()  
  
s = s[s["week"]==WEEK].copy()  
for c in ("vistm","hometm"):  
 s[c] = s[c].astype(str).str.strip().str.upper().replace({"WAS":"WSH","ARZ":"ARI","LA":"LAR"})  
  
p["\_key"] = p[["team","opponent"]].apply(lambda r: "::".join(sorted([r.team, r.opponent])), axis=1)  
s["\_key"] = s[["vistm","hometm"]].apply(lambda r: "::".join(sorted([r.vistm, r.hometm])), axis=1)  
  
# Compute PT display + numeric weekday/time key (Thu=0..Mon=4)  
from math import inf  
  
def make\_keys(row):  
 date\_raw = str(row.get("date"))  
 time\_raw = str(row.get("time"))  
 if not date\_raw or not time\_raw:  
 return pd.Series({"kickoff\_pt":"", "kickoff\_sort\_key": float("inf")})  
 # Parse date token  
 d = None  
 for dfmt in ("%m/%d/%Y", "%Y-%m-%d"):  
 try:  
 d = datetime.strptime(date\_raw.split()[0], dfmt)  
 break  
 except Exception:  
 pass  
 if d is None:  
 return pd.Series({"kickoff\_pt":"", "kickoff\_sort\_key": float("inf")})  
 # Parse time  
 t=None  
 for tfmt in ("%I:%M %p", "%H:%M"):  
 try:  
 t = datetime.strptime(time\_raw.strip(), tfmt)  
 break  
 except Exception:  
 pass  
 if t is None:  
 return pd.Series({"kickoff\_pt":"", "kickoff\_sort\_key": float("inf")})  
  
 et = datetime(d.year, d.month, d.day, t.hour, t.minute)  
 pt = et - timedelta(hours=3)  
 map7 = {3:0, 4:1, 5:2, 6:3, 0:4, 1:99, 2:99}  
 wd = map7.get(et.weekday(), 99)  
 sort\_key = float(wd\*1440 + pt.hour\*60 + pt.minute)  
 disp = pt.strftime("%I:%M %p").lstrip("0") + " PT"  
 return pd.Series({"kickoff\_pt": disp, "kickoff\_sort\_key": sort\_key})  
  
s\_keys = s.join(s.apply(make\_keys, axis=1))  
meta = s\_keys[["\_key","kickoff\_pt","kickoff\_sort\_key"]].drop\_duplicates("\_key")  
  
for col in ["kickoff\_pt","kickoff\_sort\_key","\_kickoff\_pt","\_kickoff\_sort\_key"]:  
 if col in p.columns:  
 del p[col]  
  
p = p.merge(meta, on="\_key", how="left")  
p["\_kickoff\_pt"] = p["kickoff\_pt"].replace({"": pd.NA})  
p["\_kickoff\_sort\_key"] = pd.to\_numeric(p["kickoff\_sort\_key"], errors="coerce")  
  
p.drop(columns=["\_key","kickoff\_pt","kickoff\_sort\_key"], errors="ignore", inplace=True)  
p.to\_csv(PLANNER, index=False)  
print("Kickoff helpers added →", PLANNER)  
PY

### D) Audit (structural + lines consistency)

python -m scripts.millions\_audit\_week ^  
 --season 2025 ^  
 --week 1 ^  
 --planner "picks/millions/millions\_planner.csv" ^  
 --roadmap "picks/millions/millions\_roadmap\_game.csv" ^  
 --out\_dir "picks/millions/diagnostics" ^  
 --line\_tolerance 0.5 ^  
 --spread\_range "-20,20" ^  
 --dvoa\_range "-0.6,0.6"

### E) Export member HTML (sorted by kickoff)

python scripts/millions\_export\_member\_html.py ^  
 --season 2025 ^  
 --week 1 ^  
 --planner "picks/millions/millions\_planner.csv" ^  
 --out "picks/millions/exports/member\_week\_1.html" ^  
 --sort\_by kickoff --show\_rest 0

## 5) What the member dashboard shows (today)

* **Header**: AWAY @ HOME.
* **Favorite (open/current)**: computed from \*\_spread\_home/away. If missing → TBD.
* **Circa Line**: shows planner circa\_line. If planner lacks it → TBD.
* **DVOA**: Total + Off + Def for each side (percent format with sign).
* **Badges**: Top 5 / Bottom 5 by **Total DVOA** (season‑level projection).
* **Kickoff**: always **PT**. Cards sorted **Thu → Fri → Sat → Sun → Mon (last)**.
* **Optional rest rows**: hidden by default (--show\_rest 0).

**Planned additions:**

* **Totals (Over/Under)** once open\_total/current\_total exist in roadmap → planner → exporter.
* **Line deltas** (current vs open), and **notes pills** (injury/weather) when available.

## 6) Data Health Checklist (weekly)

1. **Coverage**: Planner has 16 Week‑1 rows; no extra weeks mixed in.
2. **Missingness**: circa\_line can be blank (TBD) until contest lines land; other key fields ≤1% missing.
3. **Ranges**:
   * Spreads within [-20, 20].
   * DVOA totals within [-0.6, 0.6] (fractional; exporter formats %).
4. **Aliases**: All team/opponent codes resolve (WSH, ARI, LAR, etc.).
5. **Kickoff order**: \_kickoff\_sort\_key present; HTML sorted correctly (Mon last).
6. **Lines consistency**: roadmap vs planner within ±0.5 using the audit option.

## 7) Troubleshooting quick hits

* **PowerShell arg parsing**: If you see expected one argument for ranges, use quotes: "-20,20".
* **Zero matches on kickoff merge**: Check alias mapping for WAS→WSH, ARZ→ARI, LA→LAR, and week filter in the schedule.
* **Planner leaking weeks**: Filter to desired week and rewrite (we already fixed Week 1).
* **DVOA looks 100×**: The exporter multiplies by 100 for display; keep planner fields as **fractional** values.

## 8) How this advances the tool

* Locks a **repeatable, audited pipeline** from canon roadmap → member UI.
* Makes dashboard enhancements (totals, deltas, notes) simple schema additions.
* Gives us a **weekly runbook** that’s 100% pasteable and safe (no placeholders).

## 9) Roadmap for enhancements

* **Totals**: Add open\_total,current\_total,closing\_total,circa\_total to roadmap. Enrich planner and render in HTML (show current\_total, fallback to open if missing).
* **Circa parser**: Ingest weekly PDF and write circa\_spread\_home/away (and circa\_total when available).
* **Injury/Weather pills**: Optional small badges from roadmap columns.
* **Automation**: Batch script (or make) that runs: build → enrich → audit → export.

## 10) Mini‑reference (field meanings)

* home\_or\_away: perspective of the **team** column. Planner rows are team‑centric.
* \*\_spread\_home/away: game‑level lines from the roadmap. Exporter derives favorite.
* circa\_line: team‑oriented spread (negative = team favored). Derived from circa\_spread\_home/away until PDF ingest lands.
* team\_\*\_dvoa\_proj / opp\_\*\_dvoa\_proj: fractional DVOA projections (e.g., 0.048 = +4.8%).
* \_kickoff\_pt: display string in PT; \_kickoff\_sort\_key: numeric weekday/time order.

**End of doc.** Save this alongside the project (e.g., docs/millions\_runbook\_v1.md) and update as the schema evolves.