Data Management
DATA MANAGEMENT

UM5 ENSIAS
LECTURE ACADÉMIQUE

ENSIAS · UNIVERSITÉ MOHAMMED V DE RABAT · 1ÈRE ANNÉE · GÉNIE DE LA DATA

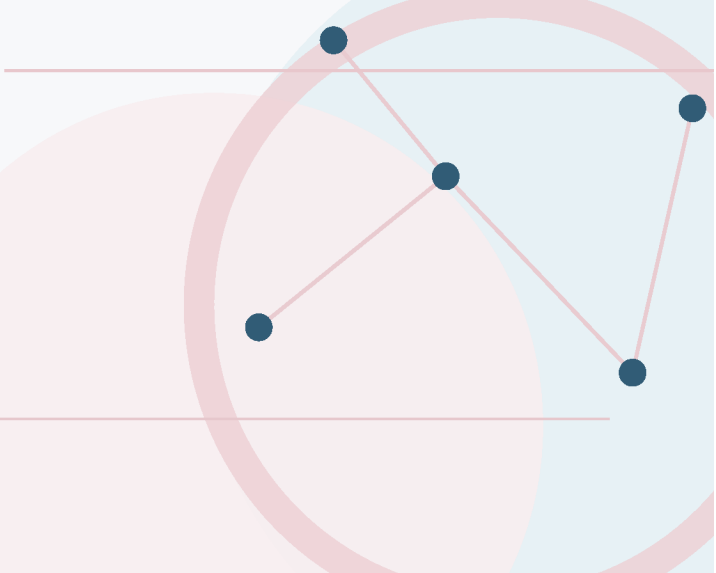
Initiation au management de la data

TP 1

De l'acquisition au nettoyage des données

De la collecte à la décision : rendre les données fiables, gouvernées et utiles

Data Management
DATA MANAGEMENT



Pr. Youness BOUTYOUR · youness.boutyour@um5.ac.ma
ENSIAS · M1151

TP · SOMMAIRE

Table des matières

Mission	3
Objectifs	3
Déroulement des 4 heures	4
Données et règles	4
Lignes de code à compléter dans le notebook	4
Questions à traiter dans le rapport	6
Préparer et exécuter	6
Livrables obligatoires	7
Contrat d’acceptation	7

ENSIAS

Cours : Initiation au management de la data · **Niveau :** 1ère année Génie de la data · **Modalité :** travail individuel · **Durée :** 4 h

Ce TP propose une activité pratique complète à partir de données fictives du campus. Vous construisez une preuve de qualité depuis la source jusqu'au jeu nettoyé.

Mission

Vous intervenez comme analyste junior pour le service énergie d'un campus. Avant de transmettre un indicateur à la direction, vous devez établir une chaîne de preuves : d'où viennent les données, que signifient-elles, quelles anomalies contiennent-elles et quelles corrections sont défendables ? La question directrice est : **peut-on comparer les consommations observées par bâtiment sans masquer les limites du dossier ?**

Objectifs

À la fin du TP, l'étudiant doit savoir :

1. charger et explorer un CSV sans modifier le brut ;
2. calculer un indicateur avec formule, unité, période et limite ;
3. acquérir une donnée par CSV, API et HTML en conservant les preuves ;
4. documenter provenance, droits, FAIR, acteurs et risques ;
5. mesurer les valeurs manquantes, doublons et valeurs invalides ;
6. classer les contrôles selon la complétude, la validité, l'unicité, la cohérence, la fraîcheur et la traçabilité ;
7. appliquer des règles de nettoyage explicites et reproductibles ;
8. produire un jeu nettoyé, un dictionnaire enrichi et un journal réutilisable.

Déroulement des 4 heures

Bloc	Durée	Production
Installation et exploration du CSV	30 min	structure et premières observations
Indicateur, cycle de vie et provenance	35 min	indicateur, provenance et analyse FAIR
API, HTML et comparaison	35 min	preuves d'acquisition et comparaison
Diagnostic qualité	25 min	contrôles avant traitement
Nettoyage et décisions	45 min	jeu nettoyé, journal et dictionnaire
Rapport et remise	30 min	archive finale et manifeste

Les 40 minutes restantes servent aux commentaires, contrôles et à la restitution.

Données et règles

Le jeu du campus contient volontairement une date absente, une consommation négative, un doublon sur `batiment` – `date`, une température manquante et des libellés à normaliser. Le fichier brut doit rester inchangé.

Règles minimales :

- date interprétable et présente ;
- bâtiment non vide après suppression des espaces ;
- consommation numérique et positive ou nulle ;
- une seule mesure par bâtiment et par date ;
- température manquante conservée et signalée.

L'appel API GitHub est limité à une requête. En cas d'indisponibilité du réseau, utiliser le fichier de repli et le signaler explicitement.

Lignes de code à compléter dans le notebook

Chaque partie pratique du notebook contient au moins une vraie tâche de programmation. Les `TODO CODE` demandent de compléter une fonction courte ou une ou plusieurs instructions Python. Les `TODO PARAMETRE` proposent en plus des valeurs simples à modifier ; les valeurs de départ restent valides.

Partie	Ligne à compléter	Résultat attendu
Partie 1 — environnement	fonction `resume_environnement`	retourner les versions utilisées
Partie 2 — observation	`COLONNE_GROUPE` et `COLONNE_MESURE`	sélectionner un groupe et une mesure numérique existants
Partie 2 — observation	fonction `compter_dimensions`	retourner le nombre de lignes et colonnes
Partie 3 — indicateur	fonction `moyenne_par_groupe`	calculer une moyenne par groupe
Partie 4 — anomalies	fonction `compter_manquantes`	compter les valeurs manquantes
Partie 5 — management	fonction `extraire_decision`	extraire une décision du vocabulaire
Partie 6 — provenance	fonction `champs_presents`	vérifier des champs obligatoires
Partie 7 — API	fonction `choisir_preuve`	choisir la preuve directe ou de repli
Partie 8 — HTML	fonction `compter_lignes_tableau`	compter les lignes extraites
Partie 9 — comparaison	fonction `trouver_mode`	filtrer un mode d'acquisition
Partie 10 — diagnostic	fonction `compter_doublons_cle`	compter les doublons d'une clé
Partie 11 — nettoyage	fonction `construire_masque_valide`	construire un masque de lignes conservables
Partie 12 — décision	fonction `creer_ligne_decision`	associer observation, preuve et limite
Partie 13 — remise	fonction `lister_sorties_absentes`	repérer les livrables manquants

Pour chaque fonction, remplacez `pass` par une ou plusieurs instructions Python, exécutez la cellule puis relancez la cellule finale de vérification. Les fonctions sont appelées par le pipeline, doivent rester courtes et ne doivent pas modifier le fichier brut. Par exemple, la fonction `compter_dimensions` doit exploiter `dataframe.shape`, `moyenne_par_groupe` doit retourner un tableau après `reset_index()`, tandis que `construire_masque_valide` doit combiner les conditions de date, de bâtiment et de consommation, puis convertir les valeurs indéterminées en `False` avec `fillna(False)`.

Les agrégations et les contrôles principaux sont fournis pour guider le travail. Les fonctions marquées `TODO CODE` sont à compléter directement dans le notebook, avec l'indice placé sous chaque fonction. La cellule finale vérifie les types, les valeurs et les livrables produits ; la dernière cellule permet d'activer un contrôle strict avant remise.

Questions à traiter dans le rapport

Répondez explicitement aux questions suivantes, en séparant le résultat observé, son interprétation et la preuve conservée :

1. Quelles colonnes, unités et périodes sont réellement disponibles dans le fichier brut ?
2. Que peut-on observer avec Pandas et que relève déjà de l'interprétation ?
3. Quel indicateur de consommation proposez-vous et quel est son dénominateur ?
4. Quelles valeurs manquantes et quels doublons sont observés avant traitement ?
5. Comment relier donnée, information, connaissance, indicateur et décision ?
6. Quels acteurs, traces, conditions d'accès et limites associer aux sources ?
7. Quelle preuve distingue une donnée acquise directement d'une donnée de repli pour l'API ? Quelle licence reste à vérifier ?
8. Quel sélecteur HTML avez-vous utilisé et quelle trace conserver si la page change ?
9. Quels sont les avantages et limites respectifs du CSV, de l'API et du HTML ?
10. Quelles anomalies concernent la clé, le domaine, la complétude ou la traçabilité ? Quelle règle appliquer à chacune ?
11. Après nettoyage, que peut-on conclure par bâtiment et que ne peut-on pas conclure ? Quelle vérification proposer avant une décision ?

Préparer et exécuter

TERMINAL · BASH

```
python3 -m venv .venv
source .venv/bin/activate
python -m pip install -r requirements.txt
python -m jupyter lab
```

Ouvrir `student/tp01_acquisition_qualite_etudiant.ipynb` depuis la racine du kit et exécuter les cellules dans l'ordre.

Le notebook contient treize complétions de code courtes, une par étape de la progression : environnement, exploration, indicateur, qualité, management, provenance, API, HTML, comparaison, diagnostic, nettoyage, décision et remise. Les valeurs proposées dans les paramètres sont déjà valides ; l'étudiant peut les modifier avec des valeurs cohérentes, puis observer le contrôle affiché.

Livrables obligatoires

L'archive finale contient `tp01_acquisition_qualite_etudiant.ipynb`, `rapport_tp01.md`, le dossier `data/` inchangé et le dossier `outputs/`.

Le dossier `outputs/` doit contenir :

- `structure_csv.json` et `controle_csv.json` ;
- `indicateur_consommation_par_batiment.csv` ;
- `provenance_tp1.json`, `api_github_pandas_observation.json` et `comparaison_sources.csv` ;
- `grille_qualite_tp1.csv` avec les dimensions et les preuves de qualité ;
- `controle_qualite_tp1.json` ;
- `donnees_nettoyees.csv` ;
- `journal_transformations.csv` ;
- `dictionnaire_donnees.csv` ;
- `synthese_decision.csv` avec observation, interprétation, limite et action ;
- `tp1_manifest.json`.

Le rapport doit distinguer observations, interprétations, incertitudes et décisions. Une donnée nettoyée sans source, règle ou preuve n'est pas considérée comme reproductible.

Contrat d'acceptation

Le TP est considéré comme terminé lorsque :

- le notebook s'exécute du début à la fin sans modifier `data/` ;
- chaque fichier produit est lisible et relié à une cellule ou une décision ;
- le diagnostic est établi avant le nettoyage ;
- le nombre de lignes avant/après et les exclusions sont justifiés ;
- le fichier nettoyé respecte les règles annoncées et reste réutilisable pour une analyse ultérieure ;
- le rapport répond à toutes les questions, contient au moins une limite et propose une action de vérification.