

Introduction

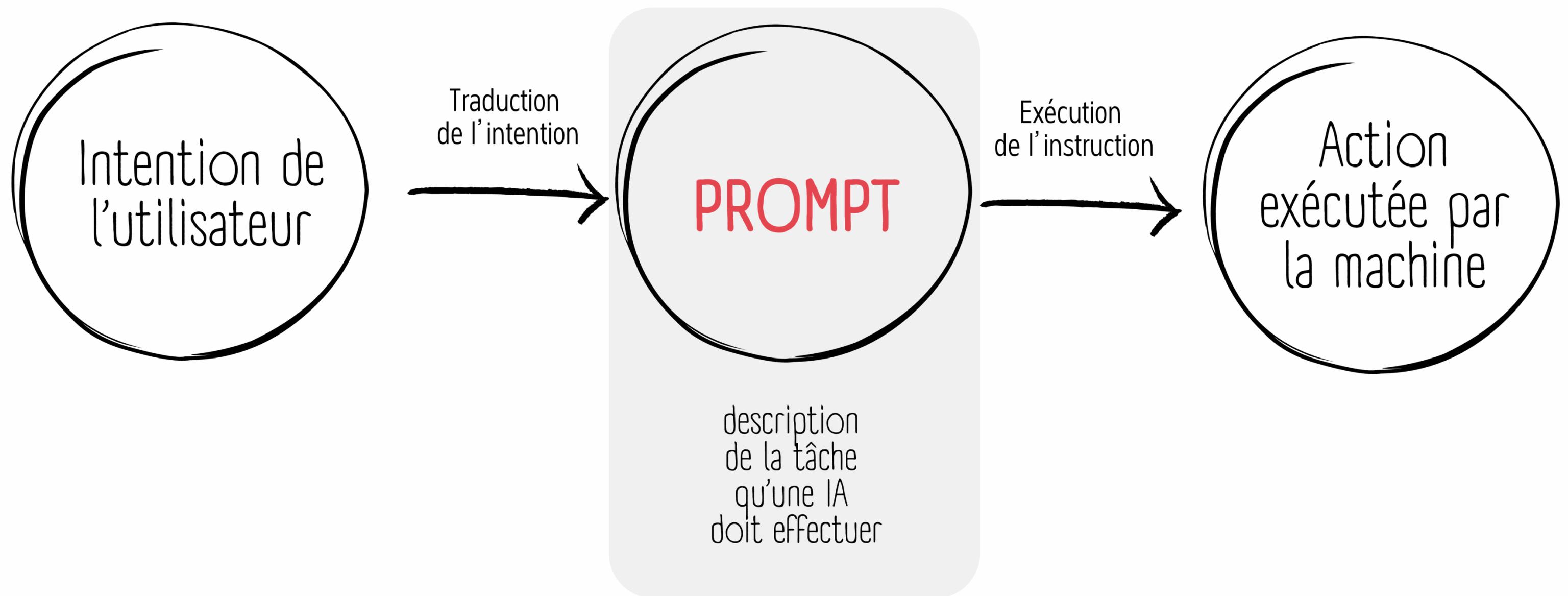
De la théorie à la *pratique*

Barbara Galimberti, Archives de la construction moderne - EPFL

L'IA au service des archives : échanges d'expériences. Demi-journée de formation du GT-DD de l'AVA, Lausanne, 31 octobre 2024

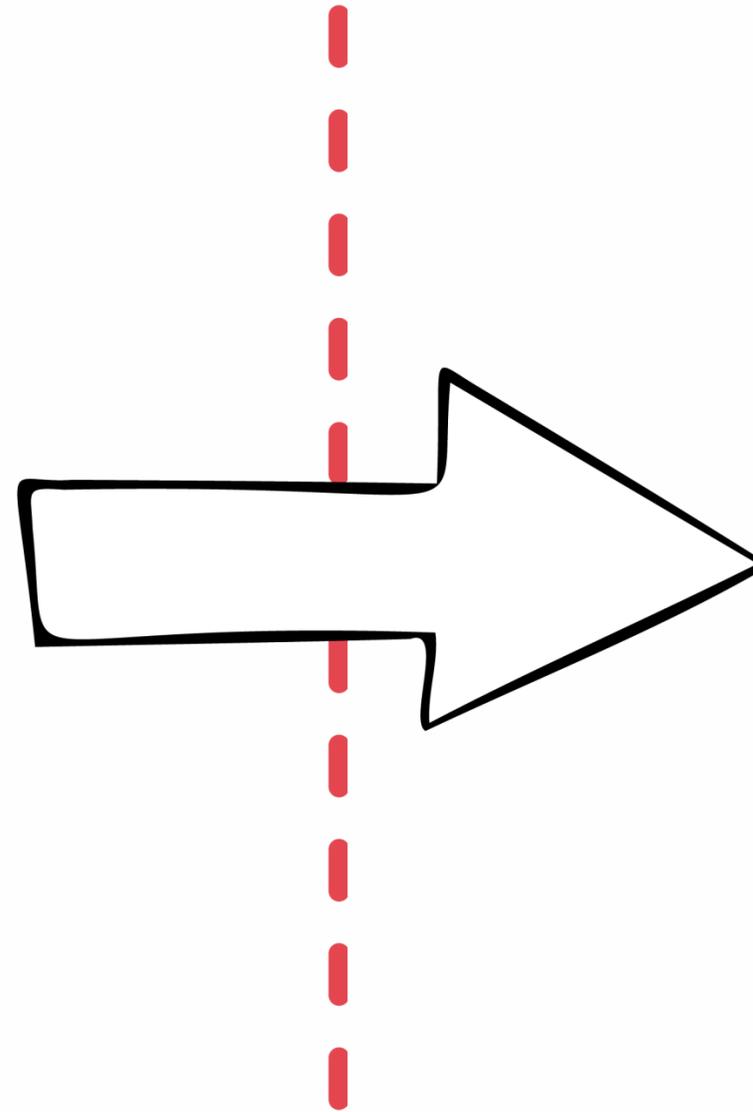
Le prompt

Un outil de simplification



Le prompt

Une rupture épistémologique



Le prompt

La construction d'un dialogue efficace

Spécificité

Détails précis
réduisant les
ambiguïtés

Contexte

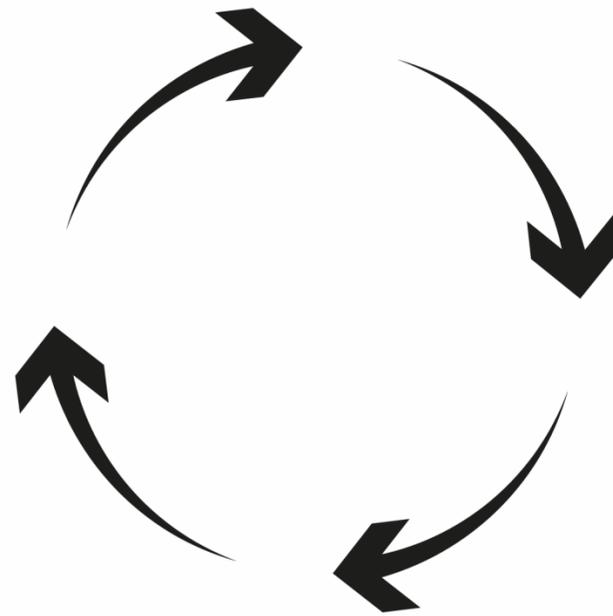
Détails
supplémentaires
pour guider l'IA

Itération

Affiner le prompt en
plusieurs essais
pour des résultats
optimaux

Retroaction

Obtenir des retours
pour mieux
comprendre les
réponses de l'IA

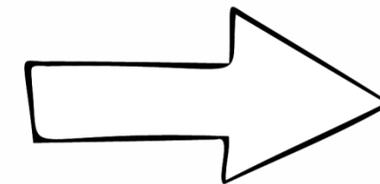


Un cas pratique

Normalisation de dates avec des formules



	A	B
1	Document ID	Date
2	001	31-10-2024
3	002	28.10.2024
4	003	2024-10-29
5	004	27/10/2024
6	005	1 Octobre 2024



jj/mm/aaaa

Dans Excel, à partir de la colonne B, ligne 2 et suivantes, les cellules contiennent des dates dans des formats variés. Crée une formule pour convertir toutes ces dates au format jj/mm/aaaa



Pour transformer les dates de différents formats en `jj/mm/aaaa` dans Excel, tu peux utiliser la fonction `TEXTE`. Voici un exemple de formule simple, à utiliser dans une nouvelle colonne à partir de la cellule B2 :

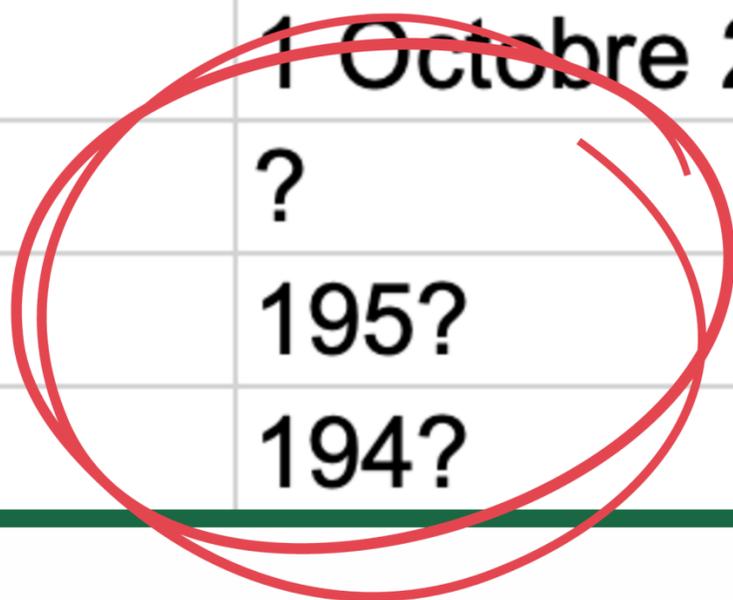
excel

Copy code

```
=TEXTE(B2;"jj/mm/aaaa")
```

	A	B	C
1	Document ID	Date	
2	001	31-10-2024	31/10/2024
3	002	28.10.2024	28/10/2024
4	003	2024-10-29	29/10/2024
5	004	27/10/2024	27/10/2024
6	005	1 Octobre 2024	01/10/2024

	A	B
1	Document ID	Date
2	001	31-10-2024
3	002	28.10.2024
4	003	2024-10-29
5	004	27/10/2024
6	005	1 Octobre 2024
7	006	?
8	007	195?
9	008	194?



Dans la colonne B, certaines cellules contiennent des dates incertaines avec un "?" à la fin, comme "195?". Crée une formule qui génère une plage de décennies (par exemple, "1950-1959" pour "195?"). Pour un simple "?", la formule doit renvoyer "s.d.". Si la date est valide, elle doit être convertie au format jj/mm/aaaa.



Pour répondre à ta demande, voici la formule qui permet de traiter les différentes situations :

excel

 Copy code

```
=SI(B2="?" ; "s.d." ;  
  SI(DROITE(B2;1)="?" ;  
    SI(ESTNUM(GAUCHE(B2;TROUVE("?" ;B2)-1)*1) ;  
      GAUCHE(B2;TROUVE("?" ;B2)-1)*1 & "0-" & GAUCHE(B2;TROUVE("?" ;B2)-1)*1 & "9"  
      "s.d.") ;  
    SI(ESTERREUR(DATEVAL(B2)) ; "s.d." ; TEXTE(DATEVAL(B2);"jj/mm/aaaa"))))
```

Explication :

1. **B2="?"** : Si la cellule contient uniquement "?", la formule renvoie "s.d."
2. **DROITE(B2;1)="?"** : Si la cellule finit par un "?", la formule interprète cela comme une date incertaine.

- Si la partie avant le "?" est un nombre (comme "195" dans "195?"), elle génère la plage

	A	B	C	D	E
1	Document ID	Date			
2	001	31-10-2024	31/10/2024		
3	002	28.10.2024	28/10/2024		
4	003	2024-10-29	29/10/2024		
5	004	27/10/2024	27/10/2024		
6	005	1 Octobre 2024	01/10/2024		
7	006	?	s.d.		
8	007	195?	1950-1959		
9	008	194?	1940-1949		

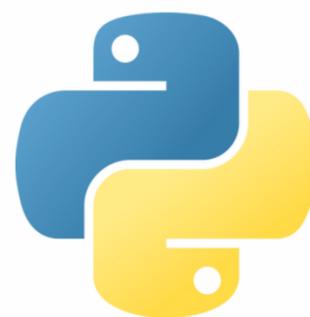
```

C2
fx =SI(B2="?", "s.d. ";
SI(DROITE(B2;1)="?";
SI(ESTNUM(GAUCHE(B2;TROUVE("?" ;B2)-1)*1);
GAUCHE(B2;TROUVE("?" ;B2)-1)*1 & "0-" & GAUCHE(B2;TROUVE("?" ;B2)-1)*1 & "9";
"s.d.");
SI(ESTERREUR(DATEVAL(B2)); "s.d."; TEXTE(DATEVAL(B2);"jj/mm/aaaa"))
)
)

```

Un cas pratique

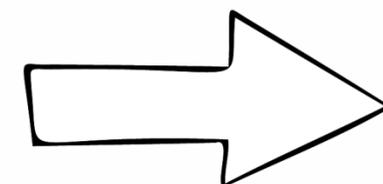
Normalisation de dates avec un script



+



	A	B
1	Document ID	Date
2	001	31-10-2024
3	002	28.10.2024
4	003	2024-10-29
5	004	27/10/2024
6	005	1 Octobre 2024
7	006	?
8	007	195?
9	008	194?



jj/mm/aaaa

J'ai un fichier Excel appelé "ClasseurTest1.xlsx" avec des dates incertaines dans la colonne "Date". Montre-moi comment utiliser Python pour lire ces données et, dans une nouvelle colonne "Date normalisée", convertir les dates en format jj/mm/aaaa. Je souhaite également que les dates incertaines, comme "1955?", soient transformées en une plage de décennies (par exemple "1950-1959"), et que les dates invalides ou incomplètes soient affichées comme "s.d.". J'aimerais être guidée étape par étape.

c/67234ec5-f13c-800d-9304-89b845a10bfc

110%



Share



python

Copy code

```
import pandas as pd
import re

# Charger le fichier Excel
file_path = "ClasseurTest1.xlsx"
df = pd.read_excel(file_path, sheet_name=0, engine='openpyxl')

# Fonction pour normaliser les dates
def normalize_date(date):
    # Vérifier si la date est de type chaîne
    if isinstance(date, str):
        # Cas où la date est incertaine, ex: "195?"
        if re.match(r'^\d{3}\?$', date):
            decade_start = date[:3] + "0" # Année de début de décenn
            decade_end = date[:3] + "9" # Année de fin de décenn
            return f"{decade_start}-{decade_end}"

        # Cas où la date est de type incomplet ou invalide
        elif not re.match(r'^\d+$', date):
            return "s.d."

    # Cas où la date est de type entier (cas typique des années complètes comme 1955)
    elif isinstance(date, int) and 1000 <= date <= 9999:
        return f"{date}-01-01"

    # Si le format est autre ou non reconnaissable, renvoie "s.d."
    return "s.d."
```



TestPython1



TestPython1

1.xlsx

2.py

Python_vide.py - /Users/acm/switchdrive/TestPython1/Python_vide.py (3.11.5)

```
df = pd.read_excel(file_path, sheet_name=0, engine='openpyxl')

# Fonction pour normaliser les dates
def normalize_date(date):
    # Vérifier si la date est de type chaîne
    if isinstance(date, str):
        # Cas où la date est incertaine, ex: "195?"
        if re.match(r'^\d{3}\?$', date):
            decade_start = date[:3] + "0" # Année de début de décennie
            decade_end = date[:3] + "9" # Année de fin de décennie
            return f"{decade_start}-{decade_end}"

        # Cas où la date est de type incomplet ou invalide
        elif not re.match(r'^\d+$', date):
            return "s.d."

        # Cas où la date est valide mais fournie en tant que chaîne
        else:
            try:
                return pd.to_datetime(date, format='%Y').strftime('%d/%m/%Y')
            except ValueError:
                return "s.d."

    # Si la date est un entier (cas typique des années complètes comme 1955)
    elif isinstance(date, int) and 1000 <= date <= 9999:
        return f"{date}-01-01"

    # Si le format est autre ou non reconnaissable, renvoie "s.d."
    return "s.d."

# Appliquer la fonction de normalisation à la colonne 'Date' et créer une nouvelle
df["Date normalisée"] = df["Date"].apply(normalize_date)
```

Conclusions

