Q1. What is Data Encoding?

Definition:
Data encoding is the process of **transforming categorical (non-numeric) data into numerical format** so that machine learning algorithms can process it. Most ML algorithms cannot handle strings or text directly, so encoding is essential.

Use in Data Science:

- Converts categorical data like "Yes/No" or "Red/Blue/Green" into numbers
- Enables algorithms like **Linear Regression, SVM, Neural Networks** to process the data
- Preserves information while making data ML-friendly

Example:

```
import pandas as pd
df = pd.DataFrame({'Color':['Red','Blue','Green']})
df['Color_encoded'] = df['Color'].map({'Red':0,'Blue':1,'Green':2})
```

Q2. What is Nominal Encoding?

Definition:
Nominal encoding assigns **unique integers** to categories in a **categorical variable with no order**.

Example (Real-world):

- Dataset column: `Fruit = ["Apple", "Banana", "Orange"]`
- Nominal encoding:
    - Apple → 0
    - Banana → 1
    - Orange → 2

```
df['Fruit_encoded'] =
df['Fruit'].map({'Apple':0,'Banana':1,'Orange':2})
```

- Use case: Encoding fruit types for predicting customer preference

---

Q3. When is Nominal Encoding Preferred over One-Hot Encoding?

Nominal encoding is preferred when:

- The categorical variable has **many unique values** (high cardinality)
- You want to **reduce dimensionality**, as one-hot encoding would create many columns

Example:

- Dataset: `City` with 100 unique cities
- One-hot encoding → 100 new columns (very sparse)
- Nominal encoding → 1 column with values 0 to 99

> **Note:** Be cautious: some algorithms may interpret integer values as ordinal, so nominal encoding is best for algorithms that **do not assume order** (like tree-based models).

Q4. Encoding a dataset with 5 unique categorical values

- Techniques:
    - **Nominal encoding**: Assign integers 0-4
    - **One-hot encoding**: Create 5 binary columns

Choice:

- If the categorical feature has **no inherent order** and **model handles integers safely** (e.g., Random Forest) → Nominal encoding
- Otherwise, use **One-hot encoding** for algorithms like Linear Regression or SVM

```
# Nominal encoding example
df['Category'] = df['Category'].map({'A':0,'B':1,'C':2,'D':3,'E':4})
```

Q5. Calculation: Nominal encoding for 2 categorical columns

- Dataset: 1000 rows, 5 columns → 2 categorical, 3 numerical
- Suppose column 1 has 3 unique categories, column 2 has 4 unique categories

Number of new columns with nominal encoding:

- Each column → 1 new column with integers
- Total = **2 new columns** (just the encoded columns themselves)

Contrast with One-hot encoding:

- Column 1 → 3 new columns
- Column 2 → 4 new columns
- Total → 7 columns

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df['Cat1_encoded'] = le.fit_transform(df['Cat1'])
df['Cat2_encoded'] = le.fit_transform(df['Cat2'])
```

Q6. Encoding for dataset with animals (species, habitat, diet)

- `Species`, `Habitat`, `Diet` → categorical variables with **no order**
- **Recommended encoding:**
    - Tree-based model → **Nominal encoding** (integer labels)
    - Linear/Distance-based model → **One-hot encoding**

```
# Example with pandas
df = pd.DataFrame({
    'Species':['Lion','Tiger','Elephant'],
    'Habitat':['Savannah','Forest','Savannah'],
    'Diet':['Carnivore','Carnivore','Herbivore']
})

df_encoded = pd.get_dummies(df)  # One-hot encoding example
```

Q7. Encoding for customer churn dataset

Dataset Features: `gender, age, contract_type, monthly_charges, tenure`

Step-by-step:

1. Identify categorical columns: `gender`, `contract_type`
2. Choose encoding:
    - `gender` → 0/1 (binary)
    - `contract_type` → One-hot encoding (if >2 categories)
3. Encode using Python:

```
import pandas as pd
```

```
from sklearn.preprocessing import LabelEncoder

# Example data
df = pd.DataFrame({
    'gender':['Male','Female','Female'],
    'contract_type':['Month-to-Month','One Year','Two Year']
})

# Gender: Binary encoding
le = LabelEncoder()
df['gender_encoded'] = le.fit_transform(df['gender'])  # Male=1,
Female=0

# Contract type: One-hot encoding
df = pd.get_dummies(df, columns=['contract_type'])
```

Result:

- `gender_encoded` → 1 column
- `contract_type` → 3 new columns (Month-to-Month, One Year, Two Year)
- Numerical columns remain unchanged
- Dataset is now **ML-ready**

✅ Summary Table

Feature	Original Type	Encoding Technique	Example
Gender	Binary categorical	Label Encoding	Male=1, Female=0
Contract type	Multi-class categorical	One-hot Encoding	Month-to-Month=1, others=0
Species	Nominal	Nominal Encoding / Label Encoding	Lion=0, Tiger=1, Elephant=2