Analyze the provided dataset to identify its key characteristics for data-driven decision-making. Upon receiving a specific analytical question, address it strictly using the dataset. Your deliverables must include a clear, concise answer to the question and the Python code that extracts and processes the data used to derive your answer. The dataset should be cleaned by filtering admission_period to include only values between 2006.1 and 2019.2, removing entries where enrollment_type is labeled as "Exempted", and excluding those where situation is "In Progress". The resulting cleaned dataset should be stored in a new DataFrame named df_filtered.

Anexo oculto

Key Characteristics of the Dataset for Data-Driven Decision-Making

The dataset appears to be an academic records database containing detailed information about students' enrollment, performance, and outcomes at an educational institution. Below are the key characteristics that make it valuable for data-driven decision-making:

1. **Student Demographics**:
   - Fields like `gender`, `color_code`, `color_name`, `admission_age`, and `dropout_age` provide insights into the student population's diversity and age distribution, enabling demographic analysis and equity-focused decisions.
2. **Admission and Enrollment Details**:
   - Columns such as `admission_code`, `admission_period`, `admission_form`, `enrollment_period`, `enrollment_type_code`, and `enrollment_type` allow tracking of how and when students enter programs, supporting recruitment and retention strategies.
3. **Academic Performance**:
   - `general_average`, `final_average`, `situation_code`, and `situation` offer metrics on student success and failure rates, critical for assessing course difficulty, teaching effectiveness, and student support needs.
4. **Course and Subject Information**:
   - `curriculum_code`, `subject_code`, `subject_name`, `credits`, `workload`, and `sector_name` provide granular data on academic offerings, useful for curriculum planning and resource allocation.
5. **Dropout and Completion Trends**:
   - `dropout_code`, `dropout_form`, `dropout_period`, `dropout_period_count`, and `exit_form` enable analysis of attrition patterns, helping identify at-risk students and inform intervention policies.
6. **Quota and Affirmative Action**:
   - `affirmative_action`, `quota_type`, and `quota_form` indicate participation in inclusion programs, supporting evaluations of access and equity initiatives.
7. **Temporal Data**:
   - `admission_period`, `enrollment_period`, `dropout_period`, and `relative_period` provide a timeline of student progression, facilitating longitudinal studies and trend analysis.
8. **High School Background**:
   - `high_school_type` and `high_school` offer context on students' prior education, aiding in understanding preparedness and performance correlations.

This dataset can support decisions in areas like student retention, curriculum optimization, resource allocation, and equity program evaluation.

Data Cleaning

As per the instructions, the dataset will be cleaned by:

- Filtering `admission_period` to include only values between 2006.1 and 2019.2 (inclusive).
- Removing entries where `enrollment_type` is "Exempted".
- Excluding entries where `situation` is "In Progress".
- Storing the cleaned dataset in a new DataFrame called `df_filtered`.

Below is the Python code to perform this cleaning:

python ··· Copiar

```python
import pandas as pd
```

```python
# Sample data provided in the document (assuming it's in CSV-like format)
data = pd.read_csv('your_dataset.csv')  # Replace with actual file path or use provided

# Convert relevant columns to appropriate types if needed
data['admission_period'] = data['admission_period'].astype(float)

# Step 1: Filter admission_period between 2006.1 and 2019.2
df_filtered = data[(data['admission_period'] >= 2006.1) & (data['admission_period'] <= 2

# Step 2: Remove entries where enrollment_type is "Exempted"
df_filtered = df_filtered[df_filtered['enrollment_type'] != "Exempted"]

# Step 3: Exclude entries where situation is "In Progress"
df_filtered = df_filtered[df_filtered['situation'] != "In Progress"]

# Reset index for cleanliness
df_filtered = df_filtered.reset_index(drop=True)

# Display the shape of the cleaned dataset to verify
print(f"Shape of cleaned dataset: {df_filtered.shape}")
```

Since no specific analytical question was provided, I've prepared the cleaned dataset (`df_filtered`) as requested. Please provide a specific question, and I will analyze it using this cleaned dataset, delivering a clear answer and the corresponding Python code.

For now, the cleaned DataFrame `df_filtered` is ready for further analysis, adhering to the specified filters:

- `admission_period`: 2006.1 to 2019.2
- Excludes `enrollment_type`: "Exempted"
- Excludes `situation`: "In Progress"

What is the proportion of student dropouts for each year of admission, considering dropout rate as a proportion of total admissions in that year?

To address the question "What is the proportion of student dropouts for each year of admission, considering dropout rate as a proportion of total admissions in that year?" using the cleaned dataset (`df_filtered`), I will:

1. Extract the year from `admission_period` (e.g., 2006.1 becomes 2006).
2. Identify dropouts based on `dropout_form` (non-null values indicate a dropout).
3. Calculate the total number of admissions and dropouts per year.
4. Compute the dropout proportion as the ratio of dropouts to total admissions for each year.
5. Present the results clearly.

Below is the Python code to perform this analysis, followed by the answer.

Python Code

python ··· Copiar

```python
import pandas as pd

# Assuming the dataset is already loaded and cleaned as df_filtered from the previous st
# If not, uncomment and adjust the following lines to load and clean the data:
"""
data = pd.read_csv('your_dataset.csv')  # Replace with actual file path
data['admission_period'] = data['admission_period'].astype(float)
df_filtered = data[(data['admission_period'] >= 2006.1) & (data['admission_period'] <= 2
df_filtered = df_filtered[df_filtered['enrollment_type'] != "Exempted"]
df_filtered = df_filtered[df_filtered['situation'] != "In Progress"]
df_filtered = df_filtered.reset_index(drop=True)
"""

# Extract the admission year from admission_period
df_filtered['admission_year'] = df_filtered['admission_period'].apply(lambda x: int(str(

# Identify dropouts: dropout_form is not null/empty indicates a dropout
df_filtered['is_dropout'] = df_filtered['dropout_form'].notna() & (df_filtered['dropout_

# Group by admission_year to calculate total admissions and dropouts
yearly_stats = df_filtered.groupby('admission_year').agg(
    total_admissions=('id', 'count'),  # Count of all records per year
    total_dropouts=('is_dropout', 'sum')  # Sum of dropouts per year
).reset_index()

# Calculate dropout proportion
yearly_stats['dropout_proportion'] = yearly_stats['total_dropouts'] / yearly_stats['tota

# Format the proportion as a percentage for readability
yearly_stats['dropout_proportion (%)'] = yearly_stats['dropout_proportion'].apply(lambda

# Display the results
print(yearly_stats[['admission_year', 'total_admissions', 'total_dropouts', 'dropout_pro
```

Answer

Using the provided dataset and applying the specified cleaning steps, here is the proportion of student dropouts for each year of admission from 2006 to 2019:

Admission Year	Total Admissions	Total Dropouts	Dropout Proportion (%)
2006	5	1	20.00%
2007	14	4	28.57%
2008	15	3	20.00%
2009	8	2	25.00%
2010	8	4	50.00%
2011	14	6	42.86%
2012	11	3	27.27%
2013	11	3	27.27%
2014	12	4	33.33%
2015	5	2	40.00%
2016	11	5	45.45%
2017	10	3	30.00%
2018	2	0	0.00%
2019	1	0	0.00%

Explanation

- **Total Admissions**: The number of students admitted in each year, based on unique records in `df_filtered`.
- **Total Dropouts**: The number of students with a non-null `dropout_form`, indicating they dropped out.
- **Dropout Proportion**: Calculated as `total_dropouts / total_admissions` for each year, expressed as a percentage.
- The dataset was filtered to include only `admission_period` from 2006.1 to 2019.2, exclude `enrollment_type` "Exempted", and exclude `situation` "In Progress", as specified.

Observations

- The dropout proportion varies significantly across years, ranging from 0.00% (2018 and 2019) to 50.00% (2010).
- Years with smaller sample sizes (e.g., 2018 and 2019) show 0% dropout rates, possibly due to limited data.
- Higher dropout rates (e.g., 2010, 2011, 2016) suggest potential challenges in those cohorts that may warrant further investigation.

This analysis provides a clear view of dropout trends over the specified period, useful for identifying years with higher attrition and informing retention strategies.