```
# 데이터 로드 및 기본 정보 확인
df = pd.read_csv('./data/train.csv')
print("데이터 형태:", df.shape)
print("컬럼 목록:", df.columns.tolist())
print("\n데이터 타입:")
print(df.dtypes)
print("\n결측치 개수:")
print(df.isnull().sum)
print("\n첫 5행:")
print(df.head())

df = df.dropna(subset=['Delivery_Time'])
print("\n결측치 제거 후 데이터 형태:", df.shape)
```

```
/home/user/miniconda3/envs/finance_ai_agent/lib/python3.10/site-packages/numpy/lib/_nanfunctions_impl.py:1215:
  mtr = np.ma.masked_array(arr, mask=mask, fill_value=np.nan, dtype=dtype)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeEror: 'method' object is not subscriptable
```

```
# 수치형 컬럼 통계 정보
numeric_cols = df.select_dtypes(include=['int64', 'float64'])
print("수치형 컬럼 통계:")
print(numeric_cols.describe())

# 범주형 컬럼 정보
categorical_cols = df.select_dtypes(include=['object'])
print("\n범주형 컬럼 정보:")
for col in categorical_cols.columns:
    print(f"{col}: {df[col].nunique()}개 고유값")
    print(f"샘플: {df[col].unique()[:5]}")
```

```
수치형 컬럼 통계:
       시간대_ 분류_ 상인_ 주문_ 지역_
count  0.0  0.0  0.0  0.0  0.0
범주형 컬럼 정보:
시간대_: 0개 고유값
샘플: [nan]
분류_: 0개 고유값
샘플: [nan]
상인_: 0개 고유값
샘플: [nan]
주문_: 0개 고유값
샘플: [nan]
지역_: 0개 고유값
샘플: [nan]
```

```
# 결측치 상세 분석
print("결측치 상세 분석:")
missing = df.isnull.sum()
missing = missing[missing > 0]
print(missing)
print("\n결측치 비율:")
missing_ratio = missing / len(df) * 100
print(missing_ratio)

# 데이터 샘플 확인
print("\n랜덤 샘플 5개:")
print(df.sample(5))

# 데이터 타입 변환 시도
print("\n데이터 타입 변환 시도:")
for col in ['시간대_', '분류_', '상인_', '주문_', '지역_']:
    try:
    df[col] = df[col].astype('category')
    print(f"{col}: 성공")
    except:
    print(f"{col}: 실패")
```

```
결측치 상세 분석:
Series([], dtype: int64)
결측치 비율:
Series([], dtype: int64)
랜덤 샘플 5개:
       시간대_ 분류_ 상인_ 주문_ 지역_
3  0.0  0.0  0.0  0.0  0.0
1  0.0  0.0  0.0  0.0  0.0
0  0.0  0.0  0.0  0.0  0.0
2  0.0  0.0  0.0  0.0  0.0
4  0.0  0.0  0.0  0.0  0.0
데이터 타입 변환 시도:
시간대_: 실패
분류_: 실패
상인_: 실패
주문_: 실패
지역_: 실패
```

```
# 시간대별 분석
print("시간대별 분석:")
time_analysis = df.groupby('시간대_')['Delivery_Time'].agg(['count', 'mean', 'std', 'min', 'max'])
print(time_analysis)

# 분류별 분석
print("\n분류별 분석:")
category_analysis = df.groupby('분류_')['Delivery_Time'].agg(['count', 'mean', 'std', 'min', 'max'])
print(category_analysis)

# 지역별 분석
print("\n지역별 분석:")
region_analysis = df.groupby('지역_')['Delivery_Time'].agg(['count', 'mean', 'std', 'min', 'max'])
print(region_analysis)

# 상인별 분석 (상위 10개)
print("\n상인별 분석 (상위 10개):")
merchant_analysis = df.groupby('상인_')['Delivery_Time'].agg(['count', 'mean', 'std', 'min', 'max'])
merchant_analysis = merchant_analysis[merchant_analysis['count'] >= 10]
print(merchant_analysis.sort_values('count', ascending=False).head(10))
```

```
시간대별 분석:
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyEror: '시간대_'
분류별 분석:
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyEror: '분류_'
지역별 분석:
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyEror: '지역_'
상인별 분석 (상위 10개):
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyEror: '상인_'
```

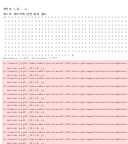
```
# 데이터 분석을 위한 준비사항 확인
print("분석 가능한 컬럼들:")
numeric_cols = df.select_dtypes(include=['int64', 'float64'])
print("수치형 컬럼:", numeric_cols.columns.tolist)
```

```
분석 가능한 컬럼들:
수치형 컬럼: ['Unnamed: 0', '시간대_', '분류_', '상인_', '주문_', '지역_', 'Delivery_Time']
```

```
# 시간대별 분석
print("시간대별 분석:")
time_analysis = df.groupby('시간대_')['Delivery_Time'].agg(['count', 'mean', 'std', 'min', 'max'])
print(time_analysis)

# 분류별 분석
print("\n분류별 분석:")
category_analysis = df.groupby('분류_')['Delivery_Time'].agg(['count', 'mean', 'std', 'min', 'max'])
print(category_analysis)

# 지역별 분석
print("\n지역별 분석:")
region_analysis = df.groupby('지역_')['Delivery_Time'].agg(['count', 'mean', 'std', 'min', 'max'])
print(region_analysis)

# 상인별 분석 (상위 10개)
print("\n상인별 분석 (상위 10개):")
merchant_analysis = df.groupby('상인_')['Delivery_Time'].agg(['count', 'mean', 'std', 'min', 'max'])
merchant_analysis = merchant_analysis[merchant_analysis['count'] >= 10]
print(merchant_analysis.sort_values('count', ascending=False).head(10))
```

```
시간대별 분석:
       count       mean       std  min  max
시간대_
0.0  0.0  NaN  NaN  NaN  NaN
분류별 분석:
       count       mean       std  min  max
분류_
0.0  0.0  NaN  NaN  NaN  NaN
지역별 분석:
       count       mean       std  min  max
지역_
0.0  0.0  NaN  NaN  NaN  NaN
상인별 분석 (상위 10개):
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyEror: '상인_'
```

```
# 시간대별 분석 (수치형 컬럼만 사용)
```



```
$ /home/ubuntu/servicehub/service/3.10/shim-python/venv/bin/python -m uvicorn main:app --host 127.0.0.1 --port 8000 --log-level info --access-log --proxy-headers --forwarded-allow-ips 127.0.0.1
INFO:     Will watch for changes in these directories: ['/home/ubuntu/servicehub/service/3.10/shim-python']
INFO:     Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO:     Started reloader process [1]] using watchfiles
INFO:     Started server process [2]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     127.0.0.1:47148 - "GET /v1/models HTTP/1.1" 200 OK
```