7．オブジェクトデータベース

1．データベーストランザクションの基礎概念

1.1．トランザクション

データベースにおける一連の処理をまとめた作業単位のことである．データの一貫性を保証するための重要な概念である．

1.2．コミット命令（commit）

トランザクションで実行した一連のデータベース操作を確定し，その結果をデータベースに永続的に反映させる命令である．

1.3．アボート命令（abort）

トランザクションで実行した一連のデータベース操作を取り消し，トランザクション開始前の状態に戻す命令である．

2．ZODBデータベースの基本操作

2.1．データベースへの接続とルートオブジェクトの取得

ZODB（Zope Object Database）は，Pythonオブジェクトを直接永続化できるオブジェクト指向データベースである．FileStorageクラスによりファイルベースのストレージを実現し，DBクラスでデータベースインスタンスを管理する．また，トランザクション管理機能も内包している．

import ZODB

import ZODB.FileStorage

storage = ZODB.FileStorage.FileStorage('hoge.fs')

db = ZODB.DB(storage)

conn = db.open()

root = conn.root()

2.2．ルートオブジェクトの確認

ZODBのルートオブジェクトはPythonの辞書型として実装されており，キーと値のペアでデータを格納する．keys()メソッドにより格納されているキーの一覧を取得できる．データベース新規作成時は空の辞書として初期化される．

print(root.keys())

2.3．トランザクションのコミット

transactionモジュールを使用してトランザクションを制御する．commit()メソッドを実行することで，メモリ上の変更内容をストレージに永続化する．これにより，トランザクションの一貫性が保証され，システム障害時のデータ保護が実現される．

import transaction

transaction.commit()

3．データ分析演習

3.1．Pandasを使用したデータ分析

3.1.1．データの読み込みと保存

Pandas（Python Data Analysis Library）のread\_csv関数を使用してCSVファイルを効率的に読み込む．header=0オプションでヘッダー行を指定し，データフレームとして構造化する．読み込んだデータはZODBのルートオブジェクトに直接格納することで永続化できる．

import pandas as pd

iris = pd.read\_csv('C:\\iris.csv', header=0)

root['iris\_data'] = iris

3.1.2．基本的な統計分析

describe()メソッドを使用して，数値データの基本統計量（平均値，標準偏差，最小値，最大値，四分位数など）を一括計算する．これによりデータの分布や外れ値を効率的に把握できる．統計量はデータ型に応じて自動的に最適化される．

# 基本統計量の計算

print(root['iris\_data'].describe())

3.1.3．データの並べ替え

Pandasのsort\_values()メソッド（旧バージョンではsort）を使用して，データを柔軟に並べ替えることができる．by引数で並べ替えの基準となる列を指定し，ascending引数で昇順（True）・降順（False）を制御する．複数列による並べ替えにも対応している．

# 昇順での並べ替え

print(root['iris\_data'].sort\_values(by='sepal\_length'))

# 降順での並べ替え

print(root['iris\_data'].sort\_values(by='sepal\_length', ascending=False))

3.1.4．頻度分析

value\_counts()メソッドを使用して，データの出現頻度を効率的に計算する．データの出現回数を自動的に集計し，デフォルトでは降順で表示する．これにより，データの分布パターンやカテゴリ別の比率を容易に把握できる．

# 頻度分布の計算

print(root['iris\_data']['sepal\_length'].value\_counts())

3.1.5．データの可視化

MatplotlibとSeabornライブラリを使用して，高度なデータの可視化を実現する．散布図により2変数間の関係性を可視化し，hueパラメータを使用してカテゴリ別に色分けして表示できる．fit\_reg=Falseオプションにより回帰直線の表示を制御できる．

# Matplotlibによる散布図（'o'は円形マーカーを指定）

plt.plot(root['iris\_data']['sepal\_length'], root['iris\_data']['sepal\_width'], 'o')

# Seabornによるカテゴリ別散布図

import seaborn

seaborn.lmplot(x='sepal\_length', y='sepal\_width', data=root['iris\_data'],

fit\_reg=False, hue='species')