試験コンテンツガイド

SAS® Viya® Intermediate Programming 試験

SAS Viya コンセプトでのプログラミング (5-10%)

SAS Viya アーキテクチャの説明

- Compute Server と Cloud Analytics Server (CAS)
- 逐次と並列処理
- インメモリ処理
- オープンソース統合

プログラミングタスクに CAS サーバーを使用する状況の説明

- データのサイズ
- 使用される SAS プロシジャの種類

CAS 対応プロシジャでのデータの管理 (10-15%)

CAS ライブラリ (caslibs)へのアクセスと使用方法の説明

- CAS ステートメントでの CAS セッションの確立
- Caslib の属性 (セッション、ローカル、アクティブ、パーソナル)
- Casuser caslib のプロパティ
- CASLIB ステートメントを使用した、セッションスコープ caslib の割り当て
- LIBNAME ステートメントと CAS エンジンを使用した caslib への libref の割り当て
- PROC CASUTIL を使用した caslib のコンテンツの表示

インメモリテーブルへのデータのロード方法の説明

- データファイルのメモリへのロード
- クライアント側とサーバー側ファイル
- クライアント側データのロード (PROC CASUTIL)
    - LOAD DATA= ステートメント
- インメモリテーブルスコープ (セッションとグローバル、テーブルのプロモート)
- サーバー側データソースのロード (PROC CASUTIL)
    - LOAD CASDATA= ステートメント
    - ALTERTABLE ステートメント

- データロードの代替方法 (DATA ステップ、PROC SQL、PROC IMPORT)

インメモリテーブルの保存と削除方法の説明

- SASHDAT ファイル
- PROC CASUTIL (SAVE と DROPTABLE ステートメント)

CAS の列データ型の説明

- 文字列の変数型のプロパティ
  - CHAR
  - VARCHAR()
  - CHAR と VARCHAR()の使い分けの判断
- 数値列の変数型のプロパティ
  - DOUBLE
  - INT32
  - INT64
- LENGTH ステートメントを使用した、varchar 列変数の作成
- 例題データの対する適切な列データ型の決定

CAS での DATA ステップと SQL プログラミング (10-15%)

SAS がコードの実行場所を決定する方法の説明

- 入力/出力データの場所
- どのようなプロシジャが実行されているか
- どのようなステートメント/関数が使用されているか
- DATA ステートメントでの SESSREF=オプション
- FedSQL での SESSREF=オプション
- MSGLEVEL= システムオプション

SAS DATA ステップでのスレッド化の説明

- コードが実行される場所：CAS、Compute Server
- DATA ステップにおけるスレッドの影響
- _THREADID_ と_NTHREADS_ 自動変数
- SINGLE= DATA ステップオプション
- 集計時の DATA ステップコードの調整

- CAS 対応 DATA ステップコードでの BY グループ処理の説明
    - スレッドの分散と BY GROUP 変数の関係
    - DATA ステップの BY GROUP 処理と並べ替え

CAS で実行する DATA ステップコードの更新

- DESCENDING キーワード
- WHERE= オプション
- INFILE/INPUT/DATALINES ステートメント
- MODIFY/REMOVE/REPLACE ステートメント
- DATALIMIT= オプション
- CAS でサポートされていない関数 (例：RANBIN、RANUNI、SYMGET、FILEREF、GIT 関数)

PROC FEDSQL コードとして実行するための PROC SQL コードの更新

- データ型
- サポートされるステートメント
- ニーモニックと演算子の比較
- SESSREF= オプション
- 再マージ
- Calculated キーワード
- SET 演算子
- 相関サブクエリ
- Dictionary テーブル
- ビュー
- LIMIT 句
- FORMAT、LABEL と PROC CASUTIL の ALTERTABLE CASDATA ステートメント

CAS 対応プロシジャとユーザー定義の出力形式 (5-10%)

Compute Server でのみ実行される共通プロシジャの識別

- PROC FREQ と UNIVARIATE
- SG Graphics プロシジャ

CAS と Compute Server の両方で実行される共通プロシジャの使用

- SAS がプロシジャの実行場所を決定する方法
    - 入力/出力データの場所
    - どの関数/オプションがコード内で使用されているか
- PROC MEANS & PROC SUMMARY
    - 共通のサポートされるステートメント：CLASS/BY/VAR/WHERE/FORMAT

  - 共通のサポートされる統計量：N、NMISS、MIN、MAX、RANGE、MEAN、SUM、STDERR、VAR)
  - 共通のサポートされない統計量：MEDIAN、MODE、パーセント点
- PROC TRANSPOSE
- CAS での BY GROUP 処理
- ログファイルを使用した、コードが実行された場所の識別

CAS でのみ実行される共通要約プロシジャの使用

- PROC FREQTAB
  - TABLE ステートメント
  - BY ステートメント
- PROC MDSUMMARY
  - VAR ステートメント
  - OUTPUT ステートメント
  - GROUPBY ステートメント

ユーザー定義の出力形式がどのように使用され、CAS に保存されるかの説明

- CAS での出力形式の保存場所
- CASFMTLIB= オプションによる caslib への出力形式の保存
- CAS ステートメントによる永続的な SASHDAT ファイルへの出力形式の保存と検索
- インメモリテーブルへの出力形式の割り当て

CAS 言語の基礎知識 (15-20%)

CASL プログラミング言語の説明

- アクションセット
- アクション
- パラメータ
- ステートメント

CASL 変数の作成と操作

- CASL 変数と SAS 変数の比較
- CASL 変数データ型 (int32、int64、double、string)
- DESCRIBE ステートメント

- PRINT ステートメント
- ビルトイン関数と共通関数の比較

CASL プログラムでの配列の使用

- 配列の定義
  - 配列要素のデータ型
  - ネストされた配列
- 配列からの値の取得
- 配列演算子 (||、&、/、==)
- 配列関数 (DIM、SORT、SORT_REV)
- 配列処理のための DO-OVER ループの使用

CASL プログラムでの辞書の使用

- 辞書の定義
- 括弧とドット表記を使用した辞書からの値の取得
- ドット表記を使用したネストされた辞書値の取得
- DO-OVER ループを使用した辞書の処理

CAS アクションから返される結果の取得

- 変数/オブジェクト/辞書として CAS アクションからの結果の取得
- アクションが成功したかどうかの確認のためリターンステータスの検証
- DO-OVER ループを使用した結果テーブルの行の処理
- 結果テーブルの保存：
  - SAVERESULT ステートメントを使用したインメモリテーブルへの保存
  - table.save アクションを使用して caslib データソースへの保存
  - SAVERESULT ステートメントを使用した SAS データセットへの保存
  - SAVERESULT ステートメントを使用した CSV ファイルへの保存

CASL プログラムでの SOURCE ブロックの使用

- コードの置換に SOURCE ウロックが必要な場合の識別
- DATA ステップと FedSQL コード置換用の SOURCE ブロックの使用
- CAS アクションの computedVarsProgram=パラメータ内での、コード置換のための SOURCE ブロックの使用

CAS アクションでの、データのアクセス (5-10%)

CAS アクションを使用したデータスへのアクセスと探索

- table.addCaslib アクションでの caslib の作成
- table.caslibInfo アクションでの利用可能な caslib 情報の表示
- table.fileInfo アクションでのデータソースファイル情報の表示
- table.loadTable アクションでのメモリへのサーバー側ファイルのロード
    - パラメータ：path、caslib、casOut、importOptions

CAS アクションを使用したインメモリテーブルの管理

- table.tableInfo アクションでのインメモリテーブル情報の表示
- table.upload アクションでのメモリへのクライアント側ファイルのロード
    - パラメータ：path、casOut
- データコネクタを使用したデータベースファイルの、メモリへのロード方法の説明
- インメモリテーブルのプロモート
- table.save でのテーブルの保存
    - パラエータ caslib= と table=
- table.dropTable アクションでのメモリからのテーブルの削除

CAS アクションでの、データの探索と検証(5-10%)

インメモリデータテーブルのプロパティとコンテンツの調査

- Table アクションセット
    - columnInfo アクション
    - fetch アクション(パラメータ table=、fetchVars=、sortBy=、from=、to=)
    - WHERE 句
- Simple アクションセット
    - numRows アクション
    - distinct アクション
- インメモリテーブル変数内の重複値の識別
    - deduplication.deduplicate アクション
- テーブルの値を期待値と比較し、ビジネスルールに従わないデータの特定

結果テーブルのプロパティとコンテンツの調査

- 結果テーブルのプロパティ値へのアクセス (nrows、ncols、name、title、attrs)
- 単一結果テーブル列からの配列の作成
- 結果テーブルコンテンツでの関数の使用 (SUM、EXISTS)
- WHERE 演算子による結果テーブルのフィルタ
- COMPUTE 演算子による計算列の作成

CAS アクションでの、データの準備 (20-25%)

table.update アクションを使用したインメモリテーブルのコンテンツの更新

- table= と set= パラメータ
  - WHERE= サブパラメータ
- set= パラメータ用の値としての辞書の配列の使用
- IFC と IFN 関数を使用した、テーブル更新時の条件付きロジックの使用
- table.update アクションの利点と考慮事項

table.copyTable アクションを使用したインメモリテーブルのコピーの作成

- 入力と出力データセットを定義する table= と casOut= パラメータ
- 列属性を設定する computedVars= パラメータ
- 列の値を設定する computedVarsProgram= パラメータ
- table.copyTable アクションの利点と考慮事項
- コピーされたテーブルのプロモート

文字列を数値列への変換

- inputn 関数での文字列を数値列への変換
- 入力形式の使用
- データ型のキャストと CAST 関数

データ準備アクションセットの使用

- dataStep アクションセット
  - runCode アクション
  - RunCodeTable アクション
- fedSQL アクションセット

    - execDirect アクション
    - CREATE TABLE、SELECT、DROP TABLE
    - query= パラメータ

table.alterTable アクションでのテーブル属性の変更

- rename=、label= パラメータでのテーブル属性の更新
- keep=、drop= パラメータでの含まれる列の変更
- columns= パラメータでの列属性の変更

dataPreprocess アクションセットを使用したテーブル内の欠損値の解決

- impute アクションを使用した欠損値の補完
    - inputs= パラメータ
    - copyAllVars= パラメータ
    - MethodInterval= と valuesInterval= パラメータ
    - 名義変数と連続変数の処理

transpose.transpose アクション を使用したテーブルの転置

- 入力と出力テーブルを指定する table= と casOut=
    - BY グループを指定する groupBy= サブパラメータ
- パラメータ: transpose、ID=、NAME=、IDLABEL=、PREFIX=

CAS アクションでの、データの分析と要約 (5-10%)

CAS アクションでのデータの要約

- simple.summary アクションでの要約統計量の生成
    - table= と casOut= パラメータ
    - inputs= パラメータ
    - subSet= パラメータ
- aggregation.aggregate アクションでの要約統計量の生成
    - table= と casOut= パラメータ
    - varSpecs= パラメータ
        - name=、subset=、agg= サブパラメータ
- dataPreprocess.rustats アクションでの要約統計量の生成
    - table=、inputs=、RequestPackages=、casOutStats= パラメータ
- 一元度数表と二元度数表の作成

- simple.freq
  - table= と inputs= パラメータ
- freqTab.freqTab
  - table= と tabulate= パラメータ
  - vars= と cross= サブパラメータ
- simple.crossTab
  - table=、row=、col=、aggregator=、weight= パラメータ

ビジュアルとレポートの作成

- CAS アクションを実行して結果テーブルの要約またはサブセットの生成
- ビジュアルプロシジャを使用した要約された結果テーブルからのグラフィックの生成
  - CSVALL、EXCEL、POWERPOINT、RTF、PDF 送信先

注意: 33 の主要な目標すべてがすべての試験でテストされます。追加の詳細では、追加の説明を行い、テストされる可能性のある全領域を定義しています。