Lecture 4 補充：CDS View 類型與實務查詢範例

編撰：國立屏東大學 周國華老師 日期：2026-03-11

一、S/4HANA 資料存取思維的轉變：從 Table 到 CDS

在 S/4HANA 時代，資料存取的核心從「直接存取實體表 (Table)」轉向「透過語意層 (CDS View) 存取」。

- **ECC 時代（Table 導向）：**開發者直接對 BKPF/BSEG 等底層表進行 SELECT，並在應用層（ABAP 程式）自行處理 Join 與聚合運算，導致效能與版本風險高。
- **S/4HANA 時代（語意層導向）：**雖然資料仍存於底層表（如 ACDOCA），但應用程式應透過 CDS View 存取。
- **Code Push-down 原則：**將繁重的運算（如 Join、聚合、過濾）交由底層 HANA 資料庫執行，僅將結果傳回應用層，實現「運算下推」。

二、CDS View 的分類與報表設計觀念

SAP 為了資料穩定性與權限治理，將 CDS View 進行結構化分類（VDM 模式：參考附錄）。開發者應根據需求選擇正確的類型：

1. **Interface View (I_ 前綴)：**
   - **定義：**SAP 定義的核心語意資料模型，直接對應核心主資料或 Universal Journal。
   - **特性：**結構完整、欄位多、偏底層，提供穩定的欄位語意。
   - **適用：**需要完整會計語意或自行掌控報表邏輯時（如：I_GLAccountLineItem）。
2. **Consumption View (C_ 前綴)：**
   - **定義：**建立在 Interface View 之上，專為「實際使用者」或 Fiori App 設計。
   - **特性：**欄位精選、具備顯示語意（如 Text、Currency），**最適合報表使用**。
3. **Private / Projection View (P_ 前綴)：**
   - 作為技術中介層使用，不建議直接用於一般業務報表。

三、【實務案例】使用 ABAP SQL 進行高效能查詢

以下展示如何以「新式 ABAP 語法」從 CDS View 撈取會計明細。此範例遵循 **「欄位裁剪 (Field Pruning)」** 與 **「條件下推」** 原則：

範例：查詢特定公司與年度的會計科目明細

ABAP 程式片段

```
" 1. 定義內部表結構 (只挑選報表需要的欄位，避免 SELECT *)
DATA: lt_gl_data TYPE TABLE OF I_GLAccountLineItem.

" 2. 執行查詢：落實 Code Push-down 原則
SELECT CompanyCode          " 公司代碼
       GLAccount            " 總帳科目
```

```
          FiscalYear                " 會計年度
          AccountingDocument    " 文件編號
          AmountInCompanyCodeCurrency " 原幣金額
          CompanyCodeCurrency    " 幣別
  FROM I_GLAccountLineItem    " 呼叫 CDS View
  INTO CORRESPONDING FIELDS OF TABLE @lt_gl_data
  WHERE CompanyCode = '1010'                    " 必備條件：限制公司範圍
      AND FiscalYear    = '2025'                    " 必備條件：限制時間區間
      AND GLAccount     BETWEEN '0011000000' AND '0011000999'.

" 3. 檢查結果
IF sy-subrc = 0.    " sy-subrc 是系統回傳碼，0 代表成功，非 0 代表失敗或異常
  WRITE: / '查詢成功，共計', lines( lt_gl_data ), '筆資料。'.
ELSE.
  WRITE: / '查無資料，請檢查權限(DCL)或輸入條件。'.
ENDIF.
```

四、效能與權限實務：避免開發地雷

開發 ABAP 報表時，語法正確不代表設計正確。請務必遵守以下原則：

1. **嚴禁 SELECT *：**在 HANA 欄式儲存架構下，選取不必要的欄位會大幅增加記憶體負擔。
2. **核心索引欄位過濾：**WHERE 條件中必須包含 CompanyCode 與 FiscalYear，以避免資料庫進行低效的全表掃描。
3. **轉義符號 @：**存取 ABAP 變數（如 lt_gl_data）時必須加上 @，這是 HANA 最佳化語法的標準要求。讓資料庫引擎知道這是一個來自應用層的變數，而非資料庫的欄位名稱。
4. **權限治理 (DCL)：**CDS View 可能內建權限檢查。若語法正確卻查無資料，通常是受使用者權限限制而非程式錯誤（Bug）。

五、總結

如果底層 ACDOCA 是存放海量食材的「大倉庫」，那麼 CDS View 就是廚師預先配好的「標準化食材」。現代化開發應直接調用這些半成品，而非重新從倉庫搬取所有原始食材，才能發揮 S/4HANA 的效能優勢。

附錄：VDM 概念解析

在 SAP S/4HANA 的開發架構中，**VDM (Virtual Data Model，虛擬資料模型)** 是一個極為重要的核心概念。簡單來說，它是一套「將複雜的底層資料庫表格，轉化為易懂且具業務意義的層次化結構」的標準。

以下是針對 VDM 模式的精簡解析

1. VDM 的核心定義

VDM 是以 **CDS Views** 為基礎所建構的層級化結構。它並不真正儲存資料，而是像一張「虛擬網」，覆蓋在混亂的底層實體表（如 ACDOCA、MARA）之上，定義了資料之間的關聯、計算與業務語意。

2. VDM 的三層標準架構

SAP 規定 VDM 必須分為以下三個主要層級，以確保系統的穩定性與重複使用性：

- **基礎層 / 介面層 (Basic / Interface Views, I_ 前綴)：**
  - **角色：**直接對接實體 Table，負責「資料清洗」。
  - **任務：**將難懂的表名與欄位名（如 MANDT, BUKRS）轉換為易懂的語意（如 Client, CompanyCode），並處理基本的 Join（如將交易檔與主檔關聯）。
- **複合層 (Composite Views, 無前綴 或用 I_、E_)：**
  - **角色：**根據特定業務主題進行「資料整合」。
  - **任務：**將多個 Interface Views 組合在一起，進行複雜的計算、聚合（加總、平均）或邏輯判斷。
- **消費層 (Consumption Views, C_ 前綴)：**
  - **角色：**最頂層，直接面對「使用者」。
  - **任務：**針對特定的 UI（如 Fiori App）或報表需求進行精選。這一層會加入顯示用的語意（例如：幣別轉換、文字描述），是**報表開發的最佳入口**。

3. 為什麼要採用 VDM 模式？

- **解耦 (Decoupling)：**即使 SAP 未來更改了底層實體表的結構，只要 VDM 的介面不變，開發者的程式或報表就不會出錯。
- **語意一致性：**確保全系統對「會計科目明細」或「庫存」的定義在各個報表中完全相同。
- **安全性：**可以在 VDM 層級直接定義 DCL (權限控管)，達到「資料列等級」的存取限制。

4. 形象化的類比

- **實體表 (Tables)：**散落在倉庫裡的原始食材（生肉、蔬菜、調味料）。
- **VDM Interface 層：**洗好、切好並分類好的食材。
- **VDM Consumption 層：**已經按照食譜配好的「半成品料理包」，廚師（開發者）拿來加熱（SELECT）即可上菜。

這套模式讓 S/4HANA 的開發從「從零開始挖資料」變成了「組合標準化組件」，大幅提升了開發效率與系統效能。