字串資料型別時注意事項：

一、Unicode 與非 Unicode 型別選擇

NVARCHAR vs VARCHAR：

- **使用 NVARCHAR**：需儲存多語言文字（如中文、日文）或特殊符號時，必須使用 `NVARCHAR` 以支援 Unicode 編碼[2]。每個字符佔用 2 bytes，儲存成本較高。
- **使用 VARCHAR**：若僅需儲存英數字元或單一語言文字（如純英文），優先選擇 `VARCHAR`（1 byte/字元），可節省 50% 儲存空間[1][2]。

定長與變長設計：

- **CHAR/NCHAR**：適用於固定長度字串（如 ISO 代碼、固定格式編號），避免空間碎片化。例如 `CHAR(10)` 儲存 3 字元時仍佔用 10 bytes。
- **VARCHAR/NVARCHAR**：適用於長度變動大的字串（如使用者名稱、地址），動態分配儲存空間，節省 30-50% 儲存[1][4]。

二、棄用型別與替代方案

- **避免 TEXT/NTEXT**：已棄用且效能低下，改用 `VARCHAR(MAX)` / `NVARCHAR(MAX)`，支援高達 2GB 資料且相容現代函數[3][4]。
- **MAX 型別謹慎使用**：若非必要儲存超大文字（如文件內容），避免使用 `VARCHAR(MAX)`，因其影響索引效能與記憶體分配[3]。

三、長度設定與索引優化

1. **合理限制長度**：
   - 明確設定 `VARCHAR(n)` 的 `n` 值（如 `VARCHAR(255)`），避免預設 `MAX` 導致不可控儲存增長。
   - 過大長度（如 `VARCHAR(4000)`）可能觸發頁面分割，影響 I/O 效能。
2. **索引設計要點**：
   - 索引鍵長度總和需 ≤ 900 bytes（`VARCHAR`）或 450 characters（`NVARCHAR`）。
   - 對長字串欄位建立索引時，考慮使用篩選索引或計算欄位縮減索引大小。

四、儲存與效能平衡策略

- **節省儲存成本**：
  - 使用 `VARCHAR` 替代 `NVARCHAR` 可降低 50% 儲存[2]。
  - 定長欄位填充空白時，評估實際資料長度分佈，避免空間浪費。
- **記憶體優化表**：
  - 在記憶體優化表中，優先使用 `BIN2` 排序規則以提升比較運算速度。

- 避免在記憶體表中使用 `NVARCHAR(MAX)`，因其強制資料儲存在離堆記憶體。

五、相容性與遷移檢查

1. **資料庫升級驗證**：
   - 從舊版 SQL Server 升級時，確認 `TEXT` 型別已轉換為 `VARCHAR(MAX)`。
   - 檢查隱含轉換風險，例如 `VARCHAR` 與 `NVARCHAR` 混合比較可能觸發轉換運算。
2. **應用層協調**：
   - 確保應用程式字串編碼與資料庫型別一致，避免因編碼轉換引發亂碼或效能損耗。

透過精準選擇字串型別、設定適當長度，並避免已棄用功能，可有效提升 SQL Server 2022 的儲存效率與查詢效能。在設計階段即需評估多語言需求、資料長度分佈及索引策略，以達到最佳實務平衡。