SAP ABAP Programozás alapjai
Beadható feladat levelező hallgatók számára

Ez a dokumentum két **választható** projektfeladatot tartalmaz, melyek célja az ABAP alapjainak gyakorlati alkalmazása az ABAP on Cloud környezetben, az ADT (ABAP Development Tools) konzol alkalmazások fejlesztésével. Mindkét feladat egy elképzelt üzleti problémát mutat be, amelynek megoldásához az órákon tanult nyelvi elemekre, DDIC ismeretekre és adatbázis-kezelésre van szükség.

Fontos: Kérjük, válasszon **egyet** a két feladat közül!

Általános Elvárások:

- **Fejlesztési Környezet**: ABAP on Cloud (gyakorlaton is használt környezet).
- **Alkalmazás Típusa**: ADT Console Applications.
- **Névkonvenció**: Minden létrehozott objektum nevében szerepeljen a NEPTUN kódja.

---

1. Projektfeladat: Hallgatói Nyilvántartás és Teljesítményelemzés

Feladat Célja

A feladat során gyakorolható a DDIC objektumok (domain, data element, adatbázis táblák) definiálása, belső táblák kezelése, alapvető ABAP SQL utasítások (SELECT, INSERT, DELETE) használata, valamint a konzolos alkalmazások fejlesztése. Különös hangsúlyt kap az adatmodell tervezése és a feltételes adatlekérdezések implementálása.

Adatbázis Koncepció: Hallgatói Nyilvántartás

Hozzon létre két adatbázis táblát az SAP Adatszótárban (DDIC) a hallgatói adatok és a teljesített tárgyaik tárolására. Minden szükséges mezőhöz hozzon létre saját DOMAIN-t és DATA ELEMENT-et.

1. **Hallgatói Fejléc Tábla (Z<NEPTUN>_STUD)**
    - **NEPTUN_CODE**: Egyedi Neptun kód (pl. CHAR 6, kulcsmező).
    - **NAME**: Hallgató neve (pl. CHAR 250).
2. **Hallgatói Tárgyak Tábla (Z<NEPTUN>_SLCT)**
    - **NEPTUN_CODE**: Neptun kód (idegen kulcs a fejléc táblára, hivatkozási integritás biztosítása).
    - **SUBJECT_CODE**: Tárgykód (pl. CHAR 10, a Neptun kód mellett a tábla kulcsmezője).
    - **CREDITS**: Tárgy kreditértéke
    - **GRADE**: Érdemjegy (pl. NUMC 1, 1-5 skálán, ahol 1 az elégtelen, 2-5 az elégséges-jeles, 0 a nem kapott aláírást).
        - Hozzon létre hozzá egy DOMAIN-t, ami csak a 0, 1, 2, 3, 4, 5 értékeket engedélyezi (0 jelenti, hogy még nincs jegy).

Alkalmazás 1: Adatbetöltő Konzol Program (Z<NEPTUN>_CL_STUDENT_DATA_LOADER)

Készítsen egy ADT konzolos programot, amely mintaadatokat tölt be a fent definiált táblákba.

- **Adattörlés**: A program indításakor törölje a **saját, feladathoz tartozó** táblák teljes tartalmát.
- **Adatbeszúrás**: Szúrjon be legalább 3-5 fiktív hallgatót a fejléc táblába az INSERT SQL utasítás segítségével.
- **Tárgyak hozzáadása**: Minden hallgatóhoz szúrjon be legalább 5-7 tárgyat az item táblába. Fontos, hogy az adatok között szerepeljenek:
    - Tárgyak, melyekből a hallgató sikeresen szerzett jegyet (grade >= 2).
    - Tárgyak, melyekből elégtelent szerzett (grade = 1).
    - Tárgyak, melyekhez még nem tartozik jegy (grade = 0).
- **Visszajelzés**: A művelet végén adjon visszajelzést a felhasználónak (konzolra írva) a sikeres adatbetöltésről, és arról, hogy hány hallgató, illetve tárgy került beszúrásra.

Alkalmazás 2: Teljesített Tárgyak Listázása Konzol Program (Z<NEPTUN>_CL_STUDENT_REPORTER)

Készítsen egy ADT konzolos programot, amely minden hallgató teljesített tárgyait listázza.

- **Lekérdezés**:

    - Minden hallgatót listázzon.
    - Minden hallgató esetén listázza azokat a tárgyait a, amelyekből legalább "elégséges" (grade >= 2) érdemjegyet szerzett.
    - A lekérdezés során használjon SELECT utasítást (lehetőleg JOIN vagy két SELECT és belső tábla feldolgozásával), és tárolja az eredményt belső táblában.

- **Kimenet**: Írja ki a hallgatók nevét, majd alatta táblázatos formában a sikeresen teljesített tárgyait (Tárgykód, Kreditérték, Érdemjegy). Rendezze a listát tárgykód szerint növekvő sorrendbe (a belső tábla rendezésével, vagy ORDER BY az SQL-ben).

  Jelölje meg azokat a hallgatókat, akik várhatóan végezni fognak (A kis adatbázisra való tekintettel már legalább 20 kreditet szereztek).

  Számítsa ki és írja ki a képernyőre az évfolyam, valamint minden hallgató kredittel súlyozott átlagát! Ebben az esetben a sikertelenül teljesített tárgyakat (grade = 1) is vegye figyelembe!

Technikai Elvárások és Alkalmazandó ABAP Elemek

- Változók, konstansok, literálok definiálása.
- Ciklusok (pl. LOOP AT, DO) és feltételes utasítások (pl. IF, ELSEIF, ELSE) használata.
- Belső táblák (standard tábla típussal, TYPE TABLE OF segítségével definiálva) használata.
- DDIC: DOMAIN, DATA ELEMENT, DATABASE TABLE létrehozása és kezelése.
- ABAP SQL: SELECT (WHERE, ORDER BY), INSERT, DELETE utasítások alkalmazása.
- Hibakezelés: SY-SUBRC ellenőrzése SQL műveletek után.

Beadandó Formátum és Dokumentáció

1. **Forráskód**: Mindkét ABAP osztály átlátható kommentekkel ellátva. A kommentek legyenek relevánsak és segítsék a kód megértését.
2. **Mérnöki Döntések Dokumentációja** (pl. .docx vagy .pdf fájlban, magyar nyelven):
    - **DDIC Objektumok Tervezése**: Írja le, hogyan tervezte meg a domaineket, data elementeket és táblákat. Indokolja meg a választott adattípusokat, hosszakat és kulcsmezőket.
    - **Felmerült Problémák és Megoldások**: Milyen kihívásokba ütközött a fejlesztés során, és hogyan oldotta meg ezeket? Milyen alternatív megoldások merültek fel, és miért a választott mellett döntött?
    - **Kódminőség**: Rövid reflexió a kódolási stílusára, olvashatóságára és a kommentek használatára.

---

2. Projektfeladat: Raktárkészlet és Mozgás Nyilvántartás

Feladat Célja

A projekt célja, hogy a hallgatók önállóan alkalmazzák az SAP ABAP alapjait egy egyszerű raktárkészlet és mozgás nyilvántartási rendszer implementálásán keresztül. A feladat során gyakorolható a DDIC objektumok (domain, data element, adatbázis táblák) definiálása, belső táblák kezelése, alapvető ABAP SQL utasítások (SELECT, INSERT, DELETE) használata, valamint a konzolos alkalmazások fejlesztése. Különös hangsúlyt kap az adatmodell tervezése és a feltételes adatlekérdezések implementálása.

Adatbázis Koncepció: Raktárkészlet és Mozgás Nyilvántartás

Hozzon létre két adatbázis táblát az SAP Adatszótárban (DDIC) a raktárkészlet adatok és a raktár mozgásainak tárolására. Minden szükséges mezőhöz hozzon létre saját DOMAIN-t és DATA ELEMENT-et.

1. **Anyagtörzs Tábla (Z<NEPTUN>_MAT)**
    - **MATERIAL_ID**: Egyedi anyagazonosító (pl. CHAR 18, kulcsmező).
    - **DESCRIPTION**: Anyag leírása (pl. CHAR 40).
    - **UNIT**: Mértékegység (pl. CHAR 3, pl. 'PC', 'KG', 'M').
2. **Raktárkészlet Mozgás Tábla (Z<NEPTUN> _MMNT)**
    - **MATERIAL_ID**: Anyagazonosító (idegen kulcs az anyagtörzs táblára, hivatkozási integritás biztosítása).
    - **MOVEMENT_ID**: Mozgás egyedi azonosítója (pl. NUMC 10, a Material ID mellett a tábla kulcsmezője).
    - **MOVEMENT_TYPE**: Mozgás típusa (pl. CHAR 1, 'I' - bevételezés, 'O' - kiadás).
        - Hozzon létre hozzá egy DOMAIN-t, ami csak az 'I' és 'O' értékeket engedélyezi.
    - **QUANTITY**: Mozgatott mennyiség (pl. DEC 13,3).
    - **MOVEMENT_DATE**: Mozgás dátuma (pl. DATS).

Alkalmazás 1: Adatbetöltő Konzol Program (Z<NEPTUN>_CL_STOCK_LOADER)

Készítsen egy ADT konzolos programot, amely mintaadatokat tölt be a fent definiált táblákba.

- **Adattörlés**: A program indításakor törölje a **saját, feladathoz tartozó** táblák teljes tartalmát.
- **Adatbeszúrás**: Szúrjon be legalább 3-5 fiktív anyagot az anyagtörzs táblába az INSERT SQL utasítás segítségével.
- **Mozgások hozzáadása**: Minden anyaghoz szúrjon be legalább 5-7 raktár mozgást az item táblába. Fontos, hogy az adatok között szerepeljenek:
    - Bevételezések (MOVEMENT_TYPE = 'I').
    - Kiadások (MOVEMENT_TYPE = 'O').
    - Olyan mozgások, amelyek egy adott anyagnál azt eredményezik, hogy a valós "készlet" nulla vagy negatív lenne (ezzel provokálva a jelentés logikáját, hogy ne csak egy szimpla összegzést alkalmazzon, hanem a valós, pozitív készletet mutassa, vagy jelezze a készlethiányt).

- **Visszajelzés**: A művelet végén adjon visszajelzést a felhasználónak a sikeres adatbetöltésről, és arról, hogy hány anyag, illetve mozgás került beszúrásra.

Alkalmazás 2: Aktuális Készlet Riport Konzol Program (Z<NEPTUN>_CL_STOCK_REPORTER)

Készítsen egy ADT konzolos programot, amely egy adott anyagra vonatkozó aktuális (pozitív) készletet mutatja.

- **Lekérdezés**:
    - Minden anyagot listázzon.
    - Számolja ki az anyag aktuális készletét: összesítse a QUANTITY értékeket úgy, hogy a MOVEMENT_TYPE = 'I' esetén hozzáadja, MOVEMENT_TYPE = 'O' esetén pedig levonja a mennyiséget.
    - **Fontos**: Csak azokat az anyagokat mutassa, amelyeknek az aggregált készlete **nagyobb, mint 0**. Ha az aggregált készlet nulla vagy negatív, jelezze a felhasználónak, hogy nincs készleten az adott anyagból, vagy készlethiány van. Ezt a jelentésben a rendelkezésre álló anyagok **után** listázza!
    - A lekérdezés során használjon SELECT utasítást (lehetőleg JOIN vagy több SELECT és belső tábla feldolgozásával), és tárolja az eredményt belső táblában.
- **Kimenet**: Írja ki az anyag leírását és mértékegységét, majd alatta az aktuális készletet.

Technikai Elvárások és Alkalmazandó ABAP Elemek

- Változók, konstansok, literálok definiálása.
- Ciklusok (pl. LOOP AT, DO) és feltételes utasítások (pl. IF, ELSEIF, ELSE) használata.
- Belső táblák (standard tábla típussal, TYPE TABLE OF segítségével definiálva) használata.
- DDIC: DOMAIN, DATA ELEMENT, DATABASE TABLE létrehozása és kezelése.
- ABAP SQL: SELECT (WHERE, SUM, ORDER BY), INSERT, DELETE utasítások alkalmazása.
- Hibakezelés: SY-SUBRC ellenőrzése SQL műveletek után.

Beadandó Formátum és Dokumentáció

1. **Forráskód**: Mindkét ABAP osztály átlátható kommentekkel ellátva. A kommentek legyenek relevánsak és segítsék a kód megértését.
2. **Mérnöki Döntések Dokumentációja** (pl. .docx vagy .pdf fájlban, magyar nyelven):
    - **DDIC Objektumok Tervezése**: Írja le, hogyan tervezte meg a domaineket, data elementeket és táblákat. Indokolja meg a választott adattípusokat, hosszakat és kulcsmezőket.
    - **Felmerült Problémák és Megoldások**: Milyen kihívásokba ütközött a fejlesztés során, és hogyan oldotta meg ezeket? Milyen alternatív megoldások merültek fel, és miért a választott mellett döntött?
    - **Kódminőség**: Rövid reflexió a kódolási stílusára, olvashatóságára és a kommentek használatára.