SAP ABAP Programozás Alapjai

Második gyakorlat – Adatmodellezés és adatbázis-lekérdezés (ABAP on Cloud)

A gyakorlaton elkészített munkáját **jegyzőkönyvezze**, amely kiterjed a meghozott fejlesztői döntések, a létrehozott adatszótár-objektumok (domain, data element, adatbázis-tábla), az elkészített forráskód, valamint a futtatás eredményeként kapott konzol kimenetek dokumentálására!

A gyakorlat témája: Termelési megrendelések nyilvántartása

Ebben a gyakorlatban egy egyszerű **termelésinformatikai modellt** épít fel, amely három adatbázis-táblára támaszkodik:

- **Termékek (gyártott cikkek)** – azok a késztermékek, amelyeket a gyár előállít.
- **Munkahelyek / gépek** – azok a gyártósorok, gépek vagy munkaállomások, ahol a termelés zajlik.
- **Gyártási megrendelések** – egy adott termék egy adott munkahelyen történő legyártására vonatkozó megrendelés.

A feladat során a hallgató:

- létrehozza a szükséges **domain** és **data element** objektumokat,
- felépíti a **három adatbázis-táblát**,
- az ABAP on Cloud trial környezetben az előre megadott teszt adatokkal **feltölti** a táblákat (a feltöltő osztály és a teszt adatok külön, másolható megoldásként kerülnek kiadásra),
- végül **ABAP SQL (SELECT)** lekérdezéseket futtat a táblákon, és azok eredményét a **konzolra** írja ki.

Elnevezési konvenció (FONTOS!)

Az ABAP on Cloud trial (ABAP for Cloud Development) környezetben minden fejlesztői objektum neve a `Z` vagy `Y` betűvel kezdődjön. Az ütközések elkerülése érdekében **minden objektum nevében szerepeljen a saját NEPTUN-kódja** az alábbi minta szerint (a példákban `<NEPTUN>` helyére a saját, 6 karakteres NEPTUN-kódja kerüljön, csupa nagybetűvel):

Objektumtípus	Minta	Példa (NEPTUN = ABC123)
Domain	`Z<NEPTUN>_D_<név>`	`ZABC123_D_MATNR`
Data element	`Z<NEPTUN>_E_<név>`	`ZABC123_E_MATNR`

Adatbázis-tábla	Z<NEPTUN>_<név>	ZABC123_PRODUCT
Osztály (class)	ZCL_<NEPTUN>_<név>	ZCL_ABC123_LOAD

0. feladat: Fejlesztői környezet előkészítése

- Jelentkezzen be az **ABAP for Cloud Development** rendszerbe az **ADT (ABAP Development Tools for Eclipse)** segítségével a gyakorlatvezető által megadott módon!
- Hozzon létre (vagy válasszon ki) egy **csomagot (package)** a saját objektumai számára.

1. feladat: Domainek létrehozása

A domain határozza meg egy mező **technikai tulajdonságait** (adattípus, hossz, tizedesjegyek, esetleg fix értékkészlet).

- Hozzon létre az alábbi domaineket a megadott technikai jellemzőkkel:

Domain neve	Adattípus	Hossz / Tizedes	Leírás
Z<NEPTUN>_D_MATNR	CHAR	10	Termékazonosító (cikkszám)
Z<NEPTUN>_D_WCID	CHAR	8	Munkahely-azonosító
Z<NEPTUN>_D_ORDNO	NUMC	10	Gyártási rendelésszám
Z<NEPTUN>_D_QTY	QUAN (DEC)	13 / 3	Mennyiség
Z<NEPTUN>_D_PRICE	DEC	11 / 2	Egységár
Z<NEPTUN>_D_STATUS	CHAR	1	Rendelés státusza (fix értékkészlettel)

- A Z<NEPTUN>_D_STATUS domainhez vegyen fel **fix értékeket (fixed values)** rövid leírással:
  - O = Nyitott (Open)
  - R = Gyártás alatt (Running)
  - C = Lezárt (Closed)

Tipp: Az ADT-ben File → New → Other ABAP Repository Object → Dictionary → Domain. A fix értékeket a domain Fixed Values szekciójában tudja megadni.

2. feladat: Data elementek létrehozása

A data element adja a mező **szemantikai jelentését** (mezőfeliratok, dokumentáció) és egy domainre vagy beépített típusra hivatkozik.

- Hozza létre az alábbi data elementeket, mindegyikhez kapcsolja a megfelelő domaint, és adjon meg rövid és közepes **mezőfeliratot (field labels)**:

Data element	Hivatkozott domain / típus	Jelentés
Z<NEPTUN>_E_MATNR	Z<NEPTUN>_D_MATNR	Cikkszám
Z<NEPTUN>_E_MAKTX	beépített SSTRING (40) vagy saját CHAR40	Termék megnevezése
Z<NEPTUN>_E_WCID	Z<NEPTUN>_D_WCID	Munkahely-azonosító
Z<NEPTUN>_E_WCTXT	CHAR (40) vagy saját domain	Munkahely megnevezése
Z<NEPTUN>_E_ORDNO	Z<NEPTUN>_D_ORDNO	Gyártási rendelésszám
Z<NEPTUN>_E_QTY	Z<NEPTUN>_D_QTY	Mennyiség
Z<NEPTUN>_E_PRICE	Z<NEPTUN>_D_PRICE	Egységár
Z<NEPTUN>_E_STATUS	Z<NEPTUN>_D_STATUS	Rendelés státusza

Tipp: Törekedjen arra, hogy a mezők jelentése egyértelmű legyen a feliratokból. A megnevezés (szöveg) típusú mezőkhöz nem feltétlenül kell saját domain – használhat beépített típust is a data element létrehozásakor.

3. feladat: Adatbázis-táblák létrehozása

Hozza létre az alábbi **három adatbázis-táblát**. Ügyeljen a kulcsmezőkre, valamint minden tábla kapjon kötelező technikai beállításokat (lásd a tippet).

3.1. Termékek tábla – Z<NEPTUN>_PRODUCT

Mező	Kulcs?	Data element / típus	Megjegyzés
client	igen	abap.clnt	Kliens mező
matnr	igen	Z<NEPTUN>_E_MATNR	Cikkszám (elsődleges kulcs)
maktx	nem	Z<NEPTUN>_E_MAKTX	Termék megnevezése
meins	nem	abap.unit(3) vagy CHAR3	Mértékegység (pl. ST, KG)
price	nem	Z<NEPTUN>_E_PRICE	Egységár

3.2. Munkahelyek tábla – Z<NEPTUN>_WORKCTR

Mező	Kulcs?	Data element / típus	Megjegyzés
client	igen	abap.clnt	Kliens mező
wcid	igen	Z<NEPTUN>_E_WCID	Munkahely-azonosító (elsődleges kulcs)
wctxt	nem	Z<NEPTUN>_E_WCTXT	Munkahely megnevezése
capacity	nem	Z<NEPTUN>_E_QTY	Napi kapacitás (db/műszak)

3.3. Gyártási megrendelések tábla – Z<NEPTUN>_PRODORD

Mező	Kulcs?	Data element / típus	Megjegyzés
`client`	igen	`abap.clnt`	Kliens mező
`ordno`	igen	`Z<NEPTUN>_E_ORDNO`	Rendelésszám (elsődleges kulcs)
`matnr`	nem	`Z<NEPTUN>_E_MATNR`	Melyik terméket gyártja (hivatkozás a Termékek táblára)
`wcid`	nem	`Z<NEPTUN>_E_WCID`	Melyik munkahelyen (hivatkozás a Munkahelyek táblára)
`qty`	nem	`Z<NEPTUN>_E_QTY`	Gyártandó mennyiség
`status`	nem	`Z<NEPTUN>_E_STATUS`	Rendelés státusza (O/R/C)
`startdate`	nem	`abap.dats`	Gyártás kezdő dátuma

Tipp: Az ADT-ben minden táblához kötelezően meg kell adni a technikai beállításokat a tábla definíciójában:

```
@EndUserText.label : 'Termékek'
@AbapCatalog.enhancement.category : #NOT_EXTENSIBLE
@AbapCatalog.tableCategory : #TRANSPARENT
@AbapCatalog.deliveryClass : #A
@AbapCatalog.dataMaintenance : #RESTRICTED
define table z<neptun>_product {
  key client : abap.clnt not null;
  key matnr  : z<neptun>_e_matnr not null;
  maktx      : z<neptun>_e_maktx;
  meins      : abap.unit(3);
  price      : z<neptun>_e_price;
}
```

A `matnr` és `wcid` mezőket a megrendelés-táblában ugyanazzal a data elementtel definiálja, mint a saját tábláikban – ez biztosítja a típus-konzisztenciát (a logikai idegenkulcs-kapcsolatot). Aktiválja mindhárom táblát!

4. feladat: A táblák feltöltése teszt adatokkal

A gyakorlaton nem kézzel viszi be az adatokat, hanem egy **ABAP osztály** futtatásával tölti fel a táblákat.

- Hozzon létre egy osztályt `ZCL_<NEPTUN>_LOAD` néven, amely az `IF_OO_ADT_CLASSRUN` interfészt implementálja (console runnable).
- Az osztály `main` metódusa:
  - törölje a három tábla korábbi tartalmát (`DELETE FROM ...`),
  - töltse fel mindhárom táblát a **gyakorlatvezető által kiadott, másolható teszt adatokkal** (`INSERT ... FROM TABLE @( VALUE #( ... ) )`),
  - írja ki a konzolra, hány rekordot szúrt be táblánként.
- Futtassa az osztályt (`F9`), és ellenőrizze a beszúrt rekordok számát!

Tipp: A teszt adatokat tartalmazó kódrészletet a gyakorlatvezető külön adja ki, amelyet bemásol az osztályba. A beszúrás után a `sy-dbcnt` rendszermező tartalmazza a legutóbbi SQL-utasítás által érintett sorok számát. Adatmódosító utasítás után ne feledje a COMMIT-ot, ha a környezet megköveteli.

Az adattáblákat a Data Preview funkcióval is ellenőrizheti az ADT-ben (jobb klikk a táblán → Open With → Data Preview).

5. feladat: Egyszerű lekérdezés – összes termék kiírása

- Hozzon létre egy új console osztályt `ZCL_<NEPTUN>_QUERY` néven (`IF_OO_ADT_CLASSRUN`).
- Olvassa ki ABAP SQL-lel a **Termékek tábla** összes sorát egy belső táblába, cikkszám szerint rendezve!
- Írja ki a konzolra szépen formázva: cikkszám, megnevezés, mértékegység, egységár.

Tipp:

```
SELECT matnr, maktx, meins, price
  FROM z<neptun>_product
  ORDER BY matnr
  INTO TABLE @DATA(lt_product).

LOOP AT lt_product INTO DATA(ls_product).
  out->write( |{ ls_product-matnr } { ls_product-maktx } { ls_product-price }|
).
ENDLOOP.
```

6. feladat: Szűrés WHERE feltétellel

- Folytassa az 5. feladatot (ugyanabban az osztályban)!
- Kérdezze le a **gyártási megrendeléseket**, amelyek státusza **nyitott (`O`)**!
- A konzolra csak a rendelésszámot, a cikkszámot és a mennyiséget írja ki.

Tipp: Használjon `WHERE status = 'O'` feltételt. A sorok száma a `lines( lt_tabla )` beépített függvénnyel kérhető le.

7. feladat: Két tábla összekapcsolása (INNER JOIN)

- Folytassa a lekérdező osztályt!
- Kapcsolja össze a **Gyártási megrendelések** és a **Termékek** táblát a `matnr` mező alapján (`INNER JOIN`), hogy a rendelésszám mellett a **termék megnevezése** is megjelenjen.
- A konzolra írja ki soronként: rendelésszám, cikkszám, **termék megnevezése**, mennyiség, státusz.

Tipp:

```
SELECT o~ordno, o~matnr, p~maktx, o~qty, o~status
  FROM z<neptun>_prodord AS o
  INNER JOIN z<neptun>_product AS p
    ON o~matnr = p~matnr
  ORDER BY o~ordno
  INTO TABLE @DATA(lt_result).
```

8. feladat: Összesítés (aggregáció, GROUP BY)

- Folytassa a lekérdező osztályt!
- Határozza meg ABAP SQL aggregációval, hogy **termékenként** összesen mekkora a megrendelt mennyiség (`SUM`), és **hány** megrendelés tartozik az adott termékhez (`COUNT`)!
- Az eredményt a **teljes megrendelt mennyiség szerint csökkenő sorrendben** írja ki a konzolra.

Tipp:

```
SELECT matnr,
       COUNT(*) AS cnt,
       SUM( qty ) AS total_qty
  FROM z<neptun>_prodord
  GROUP BY matnr
  ORDER BY total_qty DESCENDING
  INTO TABLE @DATA(lt_sum).
```

9. feladat: Háromtáblás lekérdezés és számított érték

- Folytassa a lekérdező osztályt!

- Készítsen egy lekérdezést, amely **mindhárom táblát** összekapcsolja (Gyártási megrendelések + Termékek + Munkahelyek), és soronként kiírja:
  - rendelésszám,
  - termék megnevezése,
  - munkahely megnevezése,
  - mennyiség,
  - **a megrendelés becsült értéke** = mennyiség × a termék egységára (számított mező vagy ABAP-ban kiszámítva).
- A konzolra az eredményt áttekinthető formában, oszloposan írja ki!

Tipp: A becsült érték a SELECT-ben is kiszámítható (`o~qty * p~price AS est_value`), vagy a LOOP-ban ABAP-pal. A kétszeres INNER JOIN így néz ki:

```
SELECT o~ordno, p~maktx, w~wctxt, o~qty, o~qty * p~price AS est_value
  FROM z<neptun>_prodord AS o
  INNER JOIN z<neptun>_product AS p ON o~matnr = p~matnr
  INNER JOIN z<neptun>_workctr AS w ON o~wcid  = w~wcid
  ORDER BY o~ordno
  INTO TABLE @DATA(lt_full).
```