SZAKDOLGOZAT

Katona Vilmos

Debrecen,

2011

Debreceni Egyetem

Informatikai Kar

ALKALMAZÁSFEJLESZTÉS ORACLE ESZKÖZÖKKEL

Report Groups - APEX

Konzulens:

Kollár Lajos
egyetemi tanársegéd

Készítette:

Katona Vilmos
Programtervező informatikus

Debrecen,
2011

Tartalomjegyzék

1.Bevezetés....4
2.Bevezetés az alkalmazásról....6
2.1 OBIEE – Oracle Business Intelligence Enterprise Edition....6
3.APEX – Oracle Application Express....9
3.1 Bevezetés....9
3.2 Mi is az APEX?....9
3.3 APEX történelem....10
4.Követelmények....13
5.Az alkalmazás felépítése....20
5.1 Belépési oldal....20
5.2 Fogadó/Index oldal....21
5.3 Report Group létrehozása oldal....23
5.4 Report Group módosítása oldal....24
5.5 Filterek módosítása oldal....26
5.6 Range Type Group módosítása oldal....28
6.Problémák és megoldásaik....28
6.1.1 Probléma: 20 különböző LOV megadása, kezelhetősége....28
6.1.2 A megoldás: Dinamikus LOV-ok....29
6.2.1 Probléma: 20 külön álló interaktív riport megadása és ezeken több elem egyszerre történő kiválasztása....30
6.2.2 A megoldás: Dinamikus Interaktív Riport + Checkbox + Javascript + Application Process....30
6.3 Az alkalmazás fejlesztés hátterének áttekintése....34
6.4 Trainingek....35
6.5 SDLC az NI-nál (Software Development Lifecycle)....36
7.Összefoglalás....37
8.Irodalomjegyzék....39
9.Függelék....40
10. Köszönetnyilvánítás....42
11.Plágium - Nyilatkozat....43

1.Bevezetés

2009. december végétől gyakornoki pozícióban dolgozhatok a National Instruments magyarországi részlegében, a National Instruments Hungary Kft-nél. Jelentkezésem után lehetőséget kaptam, hogy a multinacionális cég egyik erősen fejlődő, és magas üzleti értékű részlegénél az adattárház csapatába kerüljek.

Az itt töltött idő alatt több technológiával is megismerkedhettem. Az adattárház adatbázis kezelőként Oracle dbms-t használ, adatmodellező és adatintegrációs megoldásként pedig az Oracle Warehouse Builder eszközt.

010 utolsó negyedévében egy különálló alkalmazás egyszemélyes fejlesztését kezdtem el, amivel egy régi technológiával készült, nem igazán letisztult és jól működő, teljesítményileg és használhatóságilag gyenge alkalmazást fogok leváltani az új üzleti igényeket is beépítve.

A fejlesztői eszköz, amivel dolgozok az Oracle APEX (Application Express).

Amikor felvetődött a kérdés, hogy milyen témát is válasszak a szakdolgozatom számára, mindenképpen olyasmit akartam, ami közel áll hozzám, hasznos legyen és a szakma számára is megpróbáljak új és modern információkat közölni.

A szakdolgozatom egyik célja, hogy bemutassam az alkalmazásom fejlesztésének részeit, egészen a látomástól a tervezésen keresztül, néhány implementációs megoldáson át a teszteléssel és az éles rendszerre való kihelyezésével összefoglalva/egybekötve, a szoftver evolúciójáig és a képzeletbeli működésen kívül helyezéséig.

A másik cél bemutatni, hogy milyen is ez a folyamat, hogyan működik egy nagyobb cégnél. Milyen lépcsői vannak mondjuk a tervezést megelőző lépéseknek, amikor megszületik az igény egy újításra. Kik és hogyan döntenek, hogy miért kell egy ilyen alkalmazásba invesztálni.

Az alkalmazást éles környezetben fogják használni a régi változat helyett, vagyis teljes mértékben kiváltja azt, ezért fontos, hogy a régi szoftver minden funkcionalitását megtartsuk, hogy az ügyfelek, vagyis végfelhasználók ne érezzenek negatív különbségét, ne érezzék funkciók hiányát.

A feladatra a gyakornoki pozícióm nagyon megfelelőnek tűnt, úgymond szabad kezet kaptam a fejlesztésben. Az eszközt, az APEX-et nem én választottam, de a cégünknél több területen is bizonyítottan megfelelt az elvárásoknak. A követelmények feltárását már elvégezték mielőtt munkához láttam volna, s a terv egy absztrakt változata is elkészült. A döntés, hogy

én kapom a projektet az amerikai (Austin Branch, Rushika Pandya) és magyarországi managerem (Debrecen, Vágó Csaba) megbeszélése alapján született meg.

A projektbe a tervezési rész kidolgozásánál léphettem be. Az első meetingeken még megfigyelőként, de a későbbiekben, ahogy jobban megismerkedtem a témával, egyre aktívabban vettem részt. Sokat tanultam már ekkor a tapasztaltabb kollégák látásmódjait megismerve. Sok mindent átbeszéltünk, amire nem is gondoltam volna gyakornokként, kevés saját tapasztalattal, amivel rendelkeztem.

Több ötlet is felvetődött, miként tudnánk elkészíteni az alkalmazást, hogy minél több szempontból is megfeleljen az üzleti igényeknek, és persze a saját igényeinknek is.

Nagyon fontos a cégnél a szoftver minősége, annak minden alábbi jellemzőjével együtt:

- A funkcionalitás, vagyis, hogy azt a működést végezze, amit terveztünk, és úgy, ahogy terveztük.
- A megbízhatósága, ne legyenek terhelési problémák, mindig készen álljon a felhasználók kiszolgálására.
- A használhatósága, könnyen és kényelmesen lehessen felhasználni azt a felhasználók igényeinek megfelelően.
- Eredményessége és hatásfoka, vagyis a felhasználók legyenek megelégedve azzal a termékkel, amit kaptak, és a lehetőségeknek megfelelően ki tudják használni azt.
- Fenntarthatóság, vagyis a karbantartása ne okozzon komoly problémákat, vagy hosszú leállásokat, legyen megfelelően dokumentálva, és lássuk el megfelelő támogatással.
- Áttelepíthetőség, hordozhatóság, könnyen tudjuk egy másik helyen kihelyezni az szoftvert ha kell, ne okozzon nagyobb problémákat, ha mondjuk az alatta levő DBMS verziószáma megváltozik, ez ne járjon nagy költségekkel.

Egy új termék tervezésénél nem csupán a minőségi szempontokra kell figyelni, fontos, hogy tudjuk tartani a határidőket, úgymond megérje a szoftvert fejleszteni. Ezt nem csak az elején kell felmérni, hanem folyamatosan szem előtt kell tartani a fejlesztés folyamán.

Eleinte azt gondoltam, hogy nem ígérkezik túl bonyolult és összetett feladatnak, de a fejlesztés haladtával egyre jobban derültek ki az apróbb részproblémák, melyek mind megoldásra vártak, és amiket nekem kellett megoldanom.

A szakdolgozat céljai között nem szerepel az APEX mint RAD eszköz teljes mértékű bemutatása.

2.Bevezetés az alkalmazásról

Az alkalmazás neve **Report Groups**, amit magyarra a következő képen tudunk fordítani Riport/Lekérdezés/Kimutatás csoportok,halmazok.

Az alkalmazás segítségével a végfelhasználók számára lehetőség nyílik, hogy egy környezetben az általuk kívánt Grouppokat(Csoportokat), összeállíthassák, menedzselhessék vagyis kezelhessék.

- *Mi a felhasználási célja egy groupnak?*

Ahhoz, hogy ezt megértsük kezdjük azzal, hogy ezeket a termékeket, vagyis groupokat, amiket az alkalmazással előállítunk, hol használjuk fel és milyen céllal. Az eszköz, ami lehetőséget nyújt arra, hogy ezeket az összeállított groupokat, nevükkel azonosítva elérhessük röviden az OBIEE *[4]*.

2.1 OBIEE – Oracle Business Intelligence Enterprise Edition

Az Oracle Business Intelligence Enterprise Edition egy átfogó megoldást nyújt az üzleti intelligencia számára.

Széles skálán nyújt megoldásokat ezen a területen, interaktív dashboardokkal, azonnali válaszokkal különböző üzleti kérdésekre, proaktív intelligenciával a felhasználó felé irányulva és figyelmeztetésekkel.

Lehetőséget nyújt nagyvállalati és pénzügyi elemzésekhez, valósidejű előremutató értékelésekre a jelenlegi adatok alapján, emellett offline riportálásra is. Mindemellett, hogy ez a szoftver az Üzleti Intelligencia teljes tárházát adja, a platform a modern Web Service-Oriented Architektúrára épül, annak minden előnyével együtt.

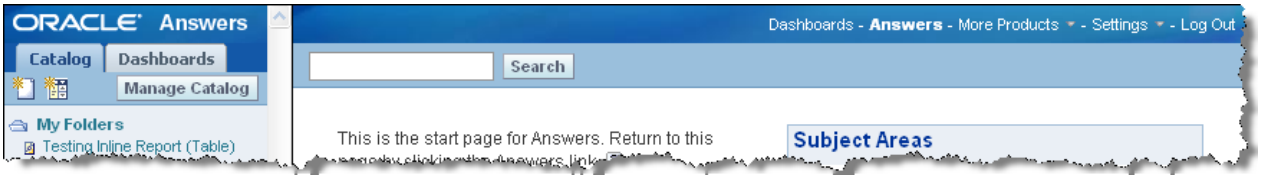
A National Instrumentsnél az OBIEE implementációját Apollo névre keresztelték. Az üzleti intelligencia számára 2009 utolsó negyedévétől vált használhatóvá, elérhetővé, így egységes riportáló (reporting tool – jelentés készítő eszköz) eszközt ad a kezükbe, ami fejlett kezelőfelülettel rendelkezik és ahogy fentebb említettem webes alapokon nyugszik.

Több szerverről, több adatbázisból képes dolgozni, képes összekapcsolni különböző üzleti területeket, így széleskörűen alkalmazható. Lehetőséget nyújt időzítésre különböző riportoknál.

A felhasználók között különálló szinteket lehet benne definiálni, ezzel elkülönítvén a különböző fontosságú, üzleti területű, vagy kritikusabb információkat.

Az eszköz végül is egy felhasználóbarát felületet nyújt lekérések megadásához, és ezek eredményeinek megjelenítéséhez grafikus chart-okon, különböző látványos és több típusú módon.

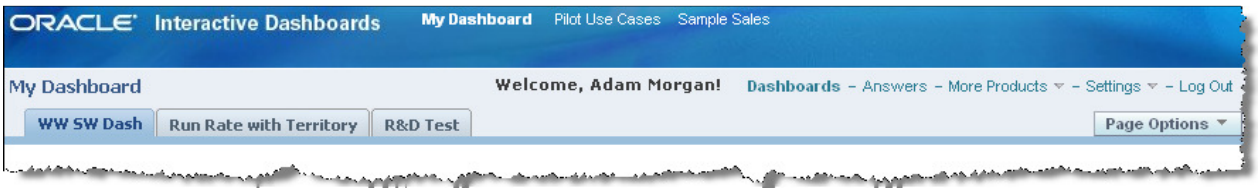
Az eszköz az NI adattárház mart(adatpiac) adatbázisához kapcsolódik, és az itt tárolt adatokat, dblinken (Database Link) keresztül éri el.



1. ábra *[5][6]*

Az 1. ábrán látható az egyik fő része az Apollonak, az Answers, ami lehetőséget nyújt:

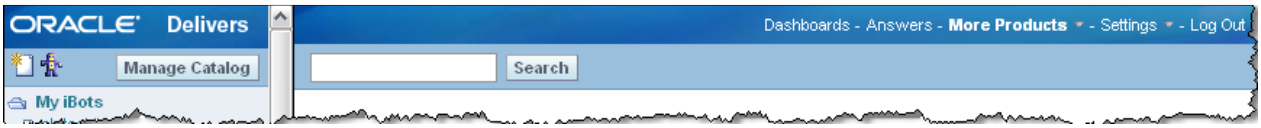
- Riportok készítésére.
- Grafikus felhasználó felületet biztosít vállalati információk elkészítésére és futtatására.
- Kérések mentésére, szervezésére, megosztására, és egyéb tartalmakkal való integrálásra.
- Az Interactive Dashboard azokat az elemeket fogja felhasználni, amelyeket az Answers-ben létrehozunk.
- Analitikus információk kinyerése és megjelenítése.



2. ábra *[5][6]*

A fenti ábrán látható a Dashboard, összevont nézet, amely lehetővé teszi az alábbiakat:

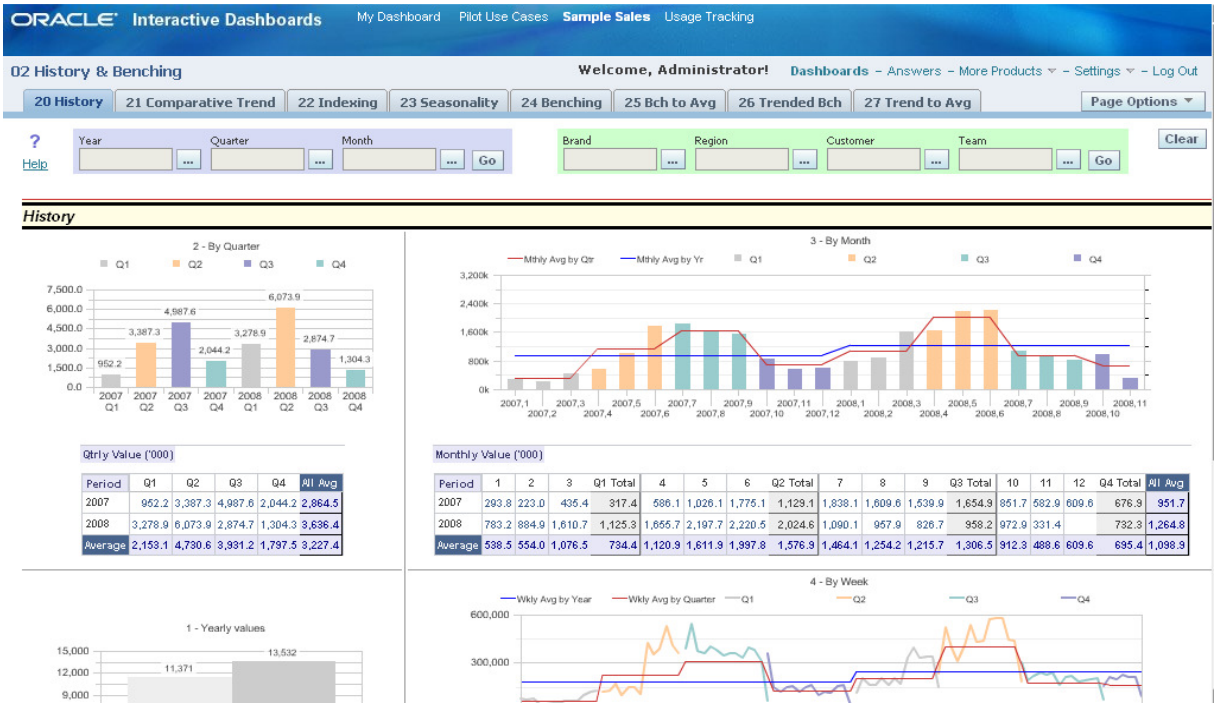
- Riportok összevonását.
- Személyre szabott nézeteket a vállalati információkról.
- Az Answers-ben készített riportokalapján dashboardok létrehozását.



3. ábra *[5][6]*

A 3. ábrán látható a Delivers, melynek funkciói az alábbiak:

- Az Apollo időzítő komponensét - másnevén iBots-ot -innen érhetjük el.
- Lehetőség nyílik az Üzleti Intelligencia feladatainak automatizálására.
- A tartalmakat megoszthatjuk több módon általa:
    - Egy linken keresztül,
    - Emailhez csatolt különféle formátumokban,
    - Smartphone-okon is elérhetővé, hozzáférhetővé tehetünk tartalmakat.
- Észrevesz különböző eredményeket jelentésekben és a célszemélyeket képes figyelmeztetni.
- Feltételekhez köthetünk különböző tartalmakat, így például kik, mikor férhetnek hozzá az adatokhoz, milyen jogosultsággal, és mekkora részéhez az információknak.



4. ábra

A 4. ábrán látható, mit lehet összerakni egy elkészült riportba, így különféle grafikonokat, számításokat, kimutatásokat. A lehetőség arra, hogy ne csak egy adott intervallumra kapjunk eredményeket, hanem ezt interaktív módon dinamikusan tovább bonthassuk további rész intervallumokra. Például Év -> Félévek -> Negyedévek.

3.APEX – Oracle Application Express

3.1 Bevezetés

Az APEX az Oracle Application Express rövidítése. Egy eszköz, amellyel lehetőség nyílik modern, web alapú alkalmazások fejlesztésére Oracle rendszereken.

Az APEX megjelenése előtt a fejlesztők számára nehézségekbe ütközött webes alkalmazások elkészítése, amelyek mögött az adatok Oracle adatbázisokban voltak tárolva. Az APEX-hez képes valóban komplex, mélyebb tapasztalatot és technológiai tudást igénylő eszközöket kellett használni, mint például NET, PHP, Java. A nehézség az integráció megoldásában rejlett, hogy miként lehet megvalósítani ezt, hogyan tudjuk közel tökéletese módon létrehozni. Ez gyakran az üzleti szabályok újraalkotásával járt az eszközök által, miközben azok már jelen voltak PL/SQL kód formájában implementálva.

Ilyen esetekben akár több hónapba és évekbe is telt, mire egy fejlesztői csapat beletanult egy adott programozási nyelvbe és megismerte a hozzátartozó technológiákat, amivel közel hatékonyan tudta megvalósítani az üzleti elvárásoknak megfelelő követelmények implementálását.

De “ezek az idők véget értek” hangzik fel z APEX-ről irt marketing szövegben.

3.2 Mi is az APEX?

Az APEX egy 100%-ban böngészőkre épülő RAD eszköz, ami lehetőséget nyújt gyors alkalmazásfejlesztésre, ezáltal gazdag, interaktív Oracle bázisú web alkalmazások elkésztésére nagyon gyorsan és viszonylag kevés programozói munkát igénybe véve.

Nagyon sok RAD(Rapid Application Development) eszköz áll rendelkezésre a különböző platformokra a piacon, de ha az adatok Oracle adatbázisokban vannak tárolva nálunk, akkor sok olyan dolog van, ami az APEX-et egyedülállóvá teszi, és elválasztja a többitől.

Először is a tény, hogy az APEX PL/SQL és SQL–re épül és mindemellett lehetőséget nyújt a nyelvek használatára, hatalmas előnyt jelent azoknak a fejlesztőknek, akik már eddig is az előző felsorolt nyelvekben fejlesztettek, hiszen minden tudásukat fel tudják használni az alkalmazások megalkotásában, ha szükséges.

Emellett, ha eddig a fejlesztőnek vagy fejlesztői csapatnak nem volt dolga velük, mindenképpen kihagyhatatlan a nyelvek megismerése. Magas prioritással az SQL nyelv alapos ismerete, de a PL/SQL ismeretek is nagyon hasznossá válhatnak a fejlesztés során.

Egy másik nagy előny az APEX mellett, hogy ez egy deklaratív eszköz, ami a magjában nagyon sok előre megírt kézre eső funkcionalitást tartalmaz, és sok olyan feladathoz, amivel külön kellene foglalkozni más web alapú fejlesztői eszköznél az APEX ehhez képest kiépített és elfedett megoldásokat tartalmaz. Ezáltal csak az alkalmazásunkhoz tartozó logika megvalósítással kell foglalkoznunk.

Az APEX beépített varázslókat (Wizards) tartalmaz. Ezek a varázslók lépésről-lépésre végig vezetik a fejlesztőt, hogy definiálhassa, mit akar majd az alkalmazása megvalósítani. Ezek után az alkalmazáshoz tartozó információkat meta-adatként tárolja.

Ahogy elértünk egy-egy ilyen varázsló végére, az alkalmazást utána szabadon módosíthatjuk, megváltoztathatunk benne az APEX által kínált megoldásokból szabadon akármennyit, hozzáadhatunk manuálisan plusz funkciókat, hatékonyabbá tehetünk különböző részeket. A fejlesztőknél nagyszámban megfigyelhető, hogy amint jobban megismerkedtek ezzel a RAD eszközzel, már nem használják a varázslókat, hanem from-scratch azaz teljes mértékben maguk állítják össze az alkalmazást.

3.3 APEX történelem

Sokan nem tudják, de az APEX már elég régóta jelen van. Az első publikus kiadás az eszközből - akkori nevén HTML DB - 2004-ben jelent meg, de a múltja ennél sokkal régebbre visszanyúlik.

Az APEX technológiai gyökerei elég régre vezethetőek vissza. Tényszerűen megállapíthatjuk, hogy a PL/SQL Web Toolkit részei, amelyek az APEX motorházteteje alatt generálják a HTML-t, melyet a böngészőnek küld a szerver, 1994-es évtől léteznek.

Ebben az időben a web alkalmazásokat PL/SQL-ben kézzel írták, ahogy a Report Groups alkalmazás ősét is, amit a szakdolgozathoz írtam át/újra az APEX-et használva. Ehhez, hogy kézzel írhassunk alkalmazásokat, nem volt elegendő a PL/SQL és HTML alapos ismerete, de rengeteg idővel, fejfájással, és türelmetlenséggel járt. A végeredmény általában egyáltalán nem felelt meg bármilyen User Interface felé hozott elvárásoknak a megjelenés terén, nem volt szép. Emellett a biztonságosságáról ne is beszéljünk. A lényeg, hogy funkcionalitásában megfelelhetett a valós elvárásoknak.

Nem sokkal ezután az Oracle bejelentette és megjelentette a PL/SQL Server Pages (PSP) névre keresztelt eszközét. Ez külön választotta a HTML statikus részét a PL/SQL kódtól egy speciális Oracle markup-ot használva, amivel meghatározhattuk, hogy hova kerül az adat.

Egy LOADPSP nevű programot használva, alakíthattuk a formázott HTML oldalunkat, a speciális Oracle markup formátumra. Ezekben az években ez egy nagy előrelépés volt, teljes keretrendszereket építettek a PSP köré.

1997-ben megjelent a WebDB, az irodalom szerint igazi nagyapja az APEX-nek. Ez egy teljes mértékben web alapú megoldást biztosított webes alkalmazásfejlesztéshez.

A WebDB segítségével egyszerűen valóra vált az, hogy meglévő adatbázisokból generálhassunk formokat, jelentéseket, grafikonokat, kalendáriumokat. Még nem volt lehetőség a sessionök (tárolt állapotok) kezelésére vagy a template(sablon) használatára, mivel a kód generálva volt, vagyis nem az eszközön keresztül és nem dinamikusan jött létre.

1999 környékén Larry Ellison megbízta Mike Hichwa-t egy feladattal, miszerint egy belső naptárkezelő alkalmazást kellene alkotnia az Oracle számára. Az eredeti ötlet az volt, hogy a meglévő WebDB-t használják fel, ami egy kezdetleges kódot fog kigenerálni, és ezt egyedi módosításokkal bővítenék a végleges alkalmazás elkészültéig. Mike Hichwa jó lehetőségnek látta ezt arra, hogy teljesen újraírja a WebDB-t valami sokkal modernebb és jobban használható eszközzé.

Mike Hichwa,Joel Kalman és Tom Kyte munkája nyomán az Oracle Flows megszületett.

Az eszköz és az alkalmazás sikerének köszönhetően 2001-ben az Oracle Flows megjelent mint termék az Oracle választékán, s az Oracle 10gR1 verzióhoz ingyenesen elérhetővé vált mint HTMLDB 1.5 kiegészítő. Időrendben:

- HTMLDB 1.6 (2004): megjelentek a témák, a master-detail formok, oldal csoportok, oldalak lezárásának lehetősége, és a nemzetközivé válhatóság bizonyos foka.

- HTMLDB 2.0 (2005): megjelent az SQL Workshop, a grafikus lekérdezés készítő, az adatbázis objektum böngésző, és a sessionök állapot védelme.

- APEX 2.2(2006): megjelentek a csomagolt alkalmazások, az APEX-hez tartozó adatszótár nézetek, és a hozzáférések kezelésére használható varázsló.

- APEX 3.0 (2007): PDF exportalás lehetővé tétele a BI Publisher-el, Microsoft Access migráció lehetősége, oldalak és régiók gyorsítótárazása.

- APEX 3.1 (2008): Bemutatták az interaktív riportot, aminek az alkalmazásomban nagy hasznát vettem, emellett a biztonságosság is nőtt.

- APEX 3.2(2009): Az Oracle Formsra épülő migrációt is segítő varázsló is elérhetővé vált, és egyéb biztonsági javítások jelentek meg.

Az alkalmazásom alatt az APEX verziószáma**: 3.1.2.00.02**

4.Követelmények

A követelményeket egy Microsoft Word dokumentum formájában tartottuk számon, melynek változásaihoz különböző verziószámokat használtunk és egy központi szerveren volt megtalálható.

A megállapodásunk szerint, ha az üzleti oldalról újabb igények érkeznek, akkor az előző verziójú dokumentumhoz nem kerül hozzáírásra, hanem egy új verziószám alatt létre kell hozni egy új dokumentumot.

5. ábra

Mielőtt elkezdeném a követelményeket leírni, szeretném megfogalmazni, adott kifejezések mit jelentenek az alkalmazás követelményeiben és az alkalmazással kapcsolatban.

Report Groups:

Az alkalmazás neve. Riportok elkészítéséhez felhasználandó információkat tartalmaz. Konkrétan egy SELECT utasítás WHERE feltételeit adhatjuk meg vele.

Report Group:

Egy egység az alkalmazásban, amelynek van egy típusa, vagyis milyen terület egyedeire adhatunk meg szűréseket, egyedek alatt értem itt az adattárházban egy egyedi azonosítókkal ellátott rekordokat. Például a termékekkel kapcsolatban kívánunk valamilyen riportot készíteni, vagy a vásárlókat szeretnénk valamilyen feltételek szerint szűrni.

Report Group Name:

Egy Report Group neve, mely egyedi azonosító.

Report Group Label:

Címke egy Report Group számára, ami azok további csoportosítását teszi lehetővé

Filterek:

A Report Group típushoz tartozó ténytáblák attribútumaiból képzett szűrési feltételek. Például egy Product (Termékek) Report Group típusban lehet egy ilyen filter hogy **part number,** ami egy termék azonosítóját jelenti. Tegyük fel egy általunk gyártott beépíthető DAQ kártyáét vagy például a Customer Locations Group típusban egy filter lehet szűrési feltétel egy adott országra, így kiszűrhetjük azokat a vásárlókat, akik onnan vásároltak, majd meghatározhatjuk, hogy például mekkora összegben.

Filter típusok:

Kétféle típussal dolgozunk: level és range. A level, vagyis szinteket meghatározó típusú filter, egy értéket tartalmaz pl.: ORSZÁG = 'MAGYAROSZÁG'.

A másik típus, ami elérhető a range, vagyis egy intervallum meghatározására használható. Ezt az alkalmazás egyelőre csak a postai irányítószámok kigyűjtésének megoldásánál használja. Például: [4031 - 4033].

A felhasználó:

Az alkalmazás végfelhasználója. Rendelkezik egy azonosítóval, ami egy felhasználó név és a hozzátartozó jelszóval, amellyel tudja csak igénybe venni az alkalmazás szolgáltatásait. Ezen kívül benne van abban a halmazban, akik számára az alkalmazás elérhető.

A végleges kialakult követelmények a következőképpen lettek megfogalmazva *[14]*:

1. Az alkamazásnak APEX-el kell elkészülnie.
2. Az alkalmazást tesztelni kell fejlesztés közben a végleges verziójáig.
3. Könnyen lehessen bővíteni új Group típusokkal.
4. Az alábbi group típusokat akarja az üzleti oldal, hogy az első verzióban elérhetővé válhassanak:
    a. Product (Termékek)
        i. Attribútumai
            1. Line
            2. Group
            3. Family
            4. Sub Family
            5. Bookings Class
            6. Part Number
        ii. A group típusa
            1. level
    b. Sales Territory(Értékesítés területei)
        i. Attribútumai
            1. VP
            2. RSM
            3. ASM
            4. DSM
        ii. A group típusa
            1. level
    c. Customer(Vásárlók)
        i. Attribútumai
            1. Account Number
            2. Organization Number
            3. Organization Name
            4. Master Organization Name

ii. A group típusa
    1. level

d. Customer Location(A vásárlók földrajzi helyzete)
    i. Attribútumai
        1. Address1
        2. Address2
        3. Address3
        4. Address4
        5. City Name
        6. Postal Code
        7. County Name
        8. Province Name
        9. State Abbrev
        10. Country Code
        11. SIC Code
    ii. A group típusa
        1. level

e. Product Interest
    i. Attribútumai
        1. Primary Interest Code
        2. Secondary Interest Code
    ii. A group típusa
        1. level

f. Geography(Földrajzi helyzet)
    i. Attribútumai
        1. Continent
        2. Region
        3. Sub-Region
        4. Country
    ii. A group típusa
        1. level

g. Segments(Szegmensek)
    i. Attribútumai
        1. Segment
        2. Sub-segment
    ii. A group típusa
        1. level

h. Stripe
    i. Attribútumai
        1. Stripe
        2. Management Stripe
        3. Substripe
        4. Bookings Class
i. Postal Code(Postai irányítószámok)
    i. Attribútmai
        1. Range-low
        2. Range-hi
        3. Country Code
    ii. A group típusa
        1. range

5. A report groupok nevei között lehessen keresni a felhasználónak.

6. Biztonság:
    a. Az alkalmazáshoz a hozzáférést a meglévő LDAP-os bejegyzéseken keresztül lehessen validálni.
    b. Ha valaki létrehoz vagy bemódosít egy report groupot, akkor azt lehessen követni, vagyis a felületen meglehessen nézni, ki hozta létre, mikor, és a későbbiekben ki módosította

7. Az alábbi attribútumok jöjjenek létre a kezdő oldalon a katalógushoz, ahol az összes létrehozott group meg fog jelenni
    a. Group Name (A group neve),
    b. Group Type (A group típusa),
    c. Group Label (egy címke, a csoportosításhoz),
    d. Created By (Ki által lett létrehozva),
    e. Creation Date (Mikor lett létrehozva),
    f. Last Updated By (Ki által lett utoljára módosítva),
    g. Last Update Date (Mikor lett módosítva utoljára).

8. A group nevét a felhasználó adhassa meg, legyen validáció, hogy elkerülhető legyen létrehozni ugyanolyan névvel még egy groupot.

9. A címke is legyen megadható a felhasználó által. Itt nincs szükség validációra, mert ennek a feladata az lesz, hogy a report groupokat csoportosíthassuk, például adott belső szervezet szerint (pr,hr,marketing,sales,production…).

10. Ha valaki létrehoz egy groupot, akkor az a bejelentkezett felhasználó saját nevével legyen ellátva, lásd Created by attribútum.

11. Funkcionalitások
    a. Létrehozás,
    b. Group módosítás,
    c. Group részleteinek megtekintése,
    d. Törlés,
        i. Az ősalkalmazásban nem volt megerősítő kérdés a felhasználói felületen, ha valaki törölni akart egy groupot, itt ez követelménnyé vált.

12. Belépési oldal,
    a. felhasználónév megadása
    b. jelszó megadás

13. Az APEX alkalmazás elkészülte után létre kell hozni a megfelelő nézeteket a marton (adatpiacon), amik lehető teszik az Apollóval(OBIEE) történő integrációt.

Ezek a követelmények természetesen változtak a fejlesztés folyamán, vagyis hol el kellett tekintenem tőlük, hol megváltoztak vagy újak kerültek közéjük, de ezeket a továbbiakban mind megemlítem az alkalmazás további tárgyalásában.

Egy fejlesztő és egy végfelhasználó teljesen másképp fogalmazhatja meg a dolgokat. A mi szervezeti felépítésünkben nagy szerepet játszanak az Üzleti Elemzők (Business Analysts). Feladatuk a kapcsolat-teremtés és a problémák tolmácsolása a két végpont a szoftver végfelhasználói vagy a szoftver megrendelői és a fejlesztők között.

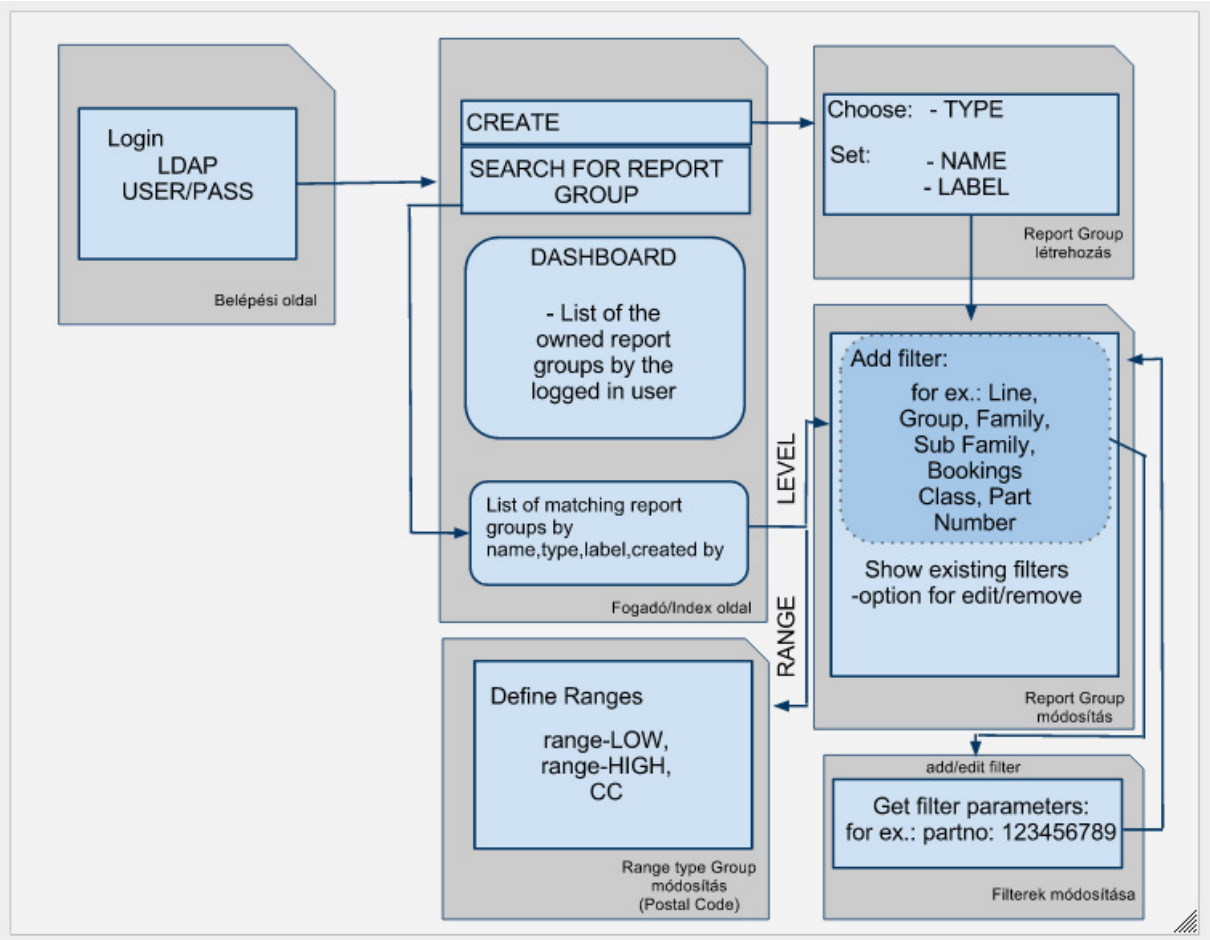
Segítségemre volt tehát egy amerikai üzleti elemző a fejlesztés folyamán. A követelményeket ő állította össze, és a Követelmények részben olvasható.

A tervezéshez és a követelmények egyeztetéséhez, pontosításához több meetingen is részt vettünk közösen egy Architect-el és a magyarországi adattárház csapat vezetőjével.

Több lehetőséget átbeszélvén a következőkre jutottunk közösen:

Az adatokat nem mozgatjuk, hozzáférésük az OBIEE-ből adatbázis linken keresztül fog történni, ami a datamart szerverünkre fog mutatni, hiszen ott találhatóak a fentebb felsorolt attribútumokat tartalmazó táblák.

A filterekből összeállított groupokat nézeteken keresztül lehet majd elérni, ahol a group nevére szűrve megkapjuk a szükséges adatokat, például a Products group típusnál ilyenkor product_id-kat fogunk visszakapni, amikből az Apolloban majd kinyerhetjük a megfelelő tényeket, például egy termék ára vagy a belőle eladott darabszám.



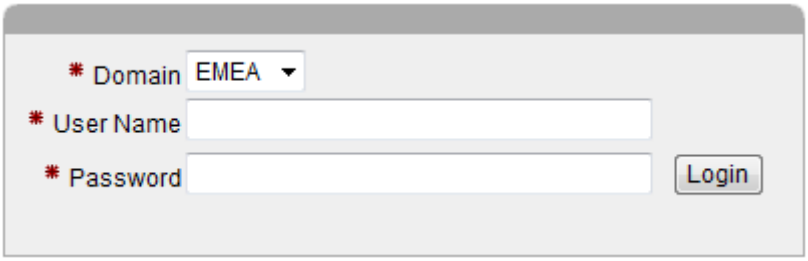
6. ábra

A 6. ábrán látható egy általam „pageflow” névre keresztelt ábra, amin az alkalmazás fő alkotórészei az oldalak és a hozzájuk tartozó funkcionális lehetőségek láthatóak valamivel érthetőbben és könnyebben.

Az előző projektjeim során megtanultam, hogy fontos mindenképpen vizuálisan felvázolni, mind saját magam, mind a megrendelői oldal számára, hogy miként tudják elképzelni a végterméket. Ezzel rengeteg problémát lehet elkerülni az elképzelések szóbeli megfogalmazásához képest. Nagyon könnyű félreérteni egymást a vevőkkel (megrendelőkkel), s e mellé még a nem anyanyelvi kommunikációból eredő esetleges félreértések is hozzáadódhatnak.

5.1 Belépési oldal

Az alkalmazás nyitó oldala a „belépési oldal”. Ez fogadja a felhasználót, amint böngészője címsorába beírja az alkalmazás eléréséhez szükséges URL-t. A belépési oldalon egy 2 inputtal rendelkező form fogja fogadni, ahol meg kell adnia a hozzáférést biztosító authentikációhoz tartozó felhasználónevét és jelszavát, amit az alkalmazás LDAP-on („Lightweight Directory Access Protocol”) keresztűl ellenőriz.



7. ábra

Mivel az alkalmazással lehetőség nyílik fontos adatokhoz tartozó lekérések összeállítására, biztonsági szempont, hogy ne férhessen hozzá bárki.

Másrészt az LDAP-os megoldást fogom arra használni, hogy kövessük a felhasználók által elvégzett változtatásokat, vagyis ki hozott létre egy groupot, ki változtatta meg annak a részleteit és mikor.

A jelenlegi követelmény-dokumentumban nem fogalmaztuk meg végül azt, hogy mit tehet egy felhasználó egy másik által létrehozott grouppal. Felvetődtek ötletek arra, hogy mit szorítsunk meg, így például ne lehessen egy másik user által létrehozott groupot törölni, vagy lehessen lockolni a sajátunkat, hogy mások ne módosíthassanak rajta.

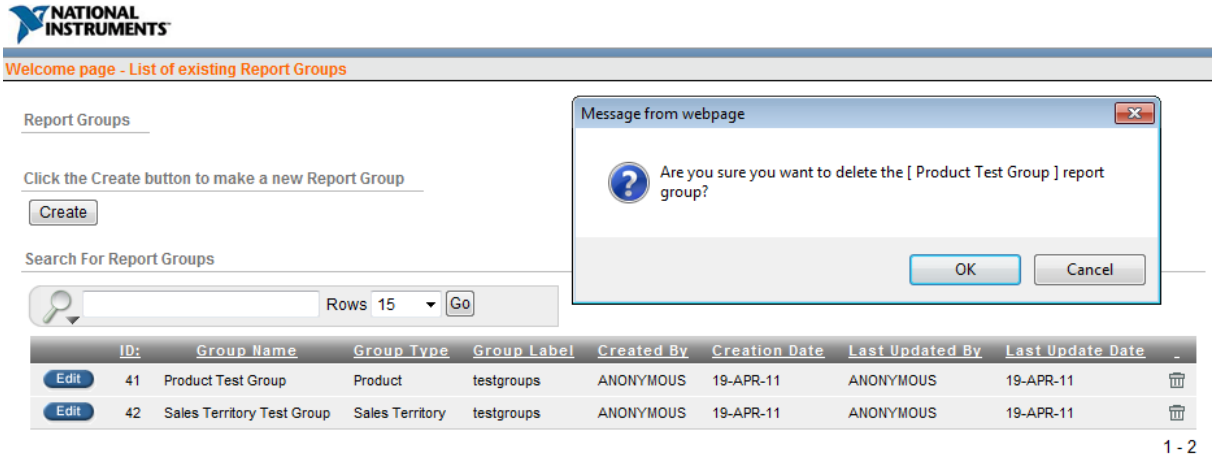
5.2 Fogadó/Index oldal

A sikeres authentikációt követvén a böngészőben a „Fogadó/Index oldal" jelenik meg, ahol a felhasználó hozzáláthat a munkához, vagyis létrehozhat új groupokat, a meglévőket törölheti, vagy különféle módosításokat végezhet el rajtuk.

Az oldalra érve az első elem, ami a felhasználói felületen a legelső helyre került a CREATE gomb, amivel elindíthatja a folyamatát egy új group elkészítésének.

A következő régióban egy interaktív riport helyezkedik el. Tartalma egy olyan lekérés eredménye, amely visszaadja a már létrehozott groupok azonosítóját és a hozzájuk tartozó leíró attribútumokat, így group nevét, típusát, címkéjét, a létrehozó felhasználó login nevét, a létrehozás dátumát, az utolsó módosító nevét és az utolsó módosítás dátumát.

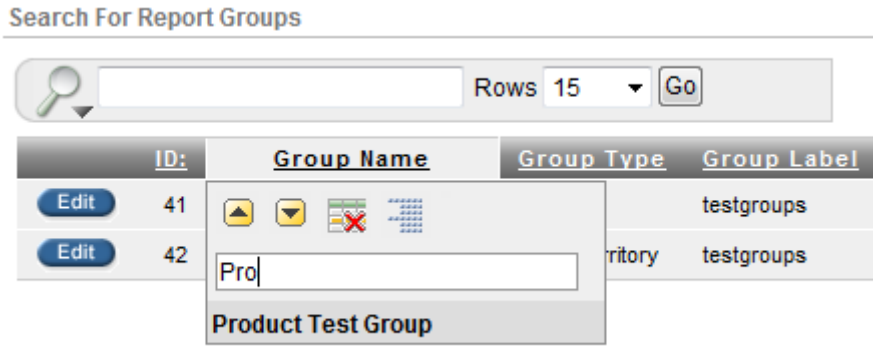
Ezeken kívül az interaktív riport tartalmaz két oszlopot, hogy az adott sorban lévő groupot módosíthassuk egy linken keresztül, és egy másik gombbal az adott sorhoz tartozó groupot kitörölhessük.



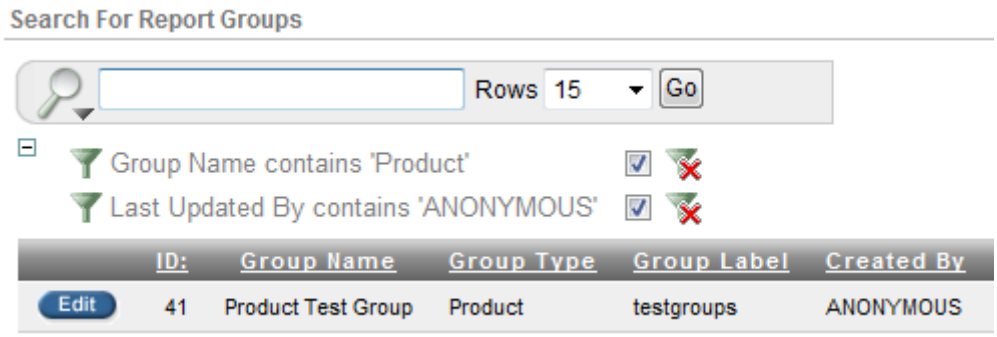
8. ábra

A törléshez a felhasználónak a jobb szélső oszlopban látható szemetes ikonra kell kattintania, ekkor egy felugró ablak kéri a művelet megerősítését. Az alkalmazás régi verziójában ilyen ellenőrzés nem történt, ezért előfordulhattak, nem kívánt report group eltávolítások, és mivel nem volt lehetőség visszaállításra, egy sok filterből álló report group is elveszhetett nagy problémákat okozva.

Az interaktív riportnak köszönhetően beépített megoldások vannak a szűrési feltételek módosításához, megadásához. Tehát például ha egy adott felhasználó nevére szeretnénk szűrni, vagy a létrehozási dátum szerint csökkenő sorrendben kívánjuk megjeleníteni az adatokat, nem kell mást tennünk, mint a riport fejlécében található attribútumokra kattintva beállítani a rendezési irányt vagy megadni a szűrési feltételt. Egy másik módszer az, ha a nagyító ikonra kattintunk, kiválaszthatjuk, melyik attribútumokra kívánunk megadni feltételeket.



9. ábra



10. ábra

Később lehet, hogy lesz egy új kérés az üzleti oldalról egy dashboard elkészítésére. Lényegében azt jelentené a dashboard, hogy a felhasználó kiteheti a számára gyakrabban módosított report groupokat egy másik régióban.

Erre egy újabb interaktív riportot kell majd elkészíteni, és persze valamilyen módon el kell tárolni az adatbázisban a favorizált riport groupokat felhasználónként.

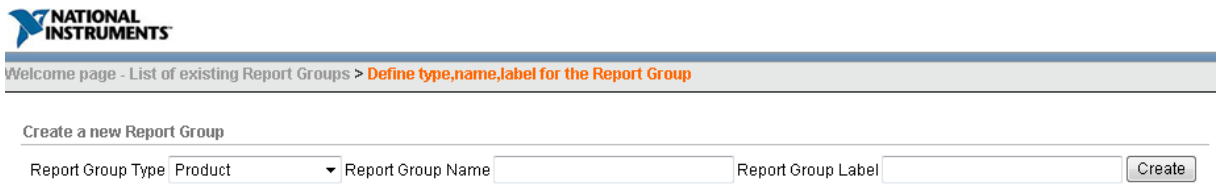
Erről az oldalról a következő műveletekkel haladhatunk tovább:

1. „CREATE” gomb: A gombra kattintva hozzákezdhetünk egy új report group létrehozásának folyamatához.
2. Az interaktív riport bal szélső oszlopában az EDIT gombra kattintva szerkeszthetjük az adott riportot.

A fenti két lehetőség két külön oldal APEX-ben, vagyis több változó értékét is át kell adnunk a számukra, hogy annak megfelelően tudják az adatok betölteni és szerkeszthetővé tenni.

5.3 Report Group létrehozása oldal

Egy report group létrehozása az index oldalon a CREATE gomb megnyomása utána ezen az oldalon folytatódik.



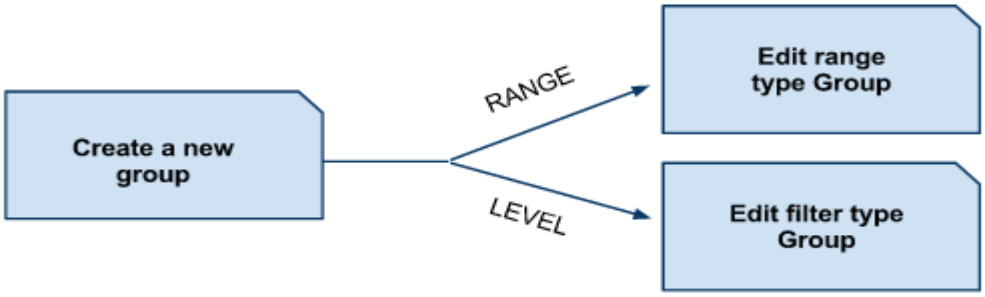
11. ábra

Az oldal lényeges része a form, ami három elemet tartalmaz. Az első elem egy lenyitható LOV(List Of Values), itt választhatjuk ki az új report group típusát.

A második elem egy input mező, ahol megadhatjuk a report grouphoz tartozó nevet, ami nem lehet megegyező az eddig létrehozottakkal, erre az adott oldalon egy validáció van beállítva, amely ellenőrzi, hogy vajon az adatbázisban szerepel-e már egy ilyen nevű group.

A harmadik elem a Label vagyis címke, ami a csoportosításhoz vagy egyszerűen bővebb információ megadására használható. Ha nincs validáció, lehet azonos címkéket különböző report groupnak megadni. A „Create” gomb megnyomása után elkészül egy a fenti attribútumokkal megadott report group és a böngésző a „report group módosítása” oldalra navigál, kivéve ha egy range típusú report groupot hoztunk létre, ekkor a „Range type Group

módosítása” oldalra kerülünk.

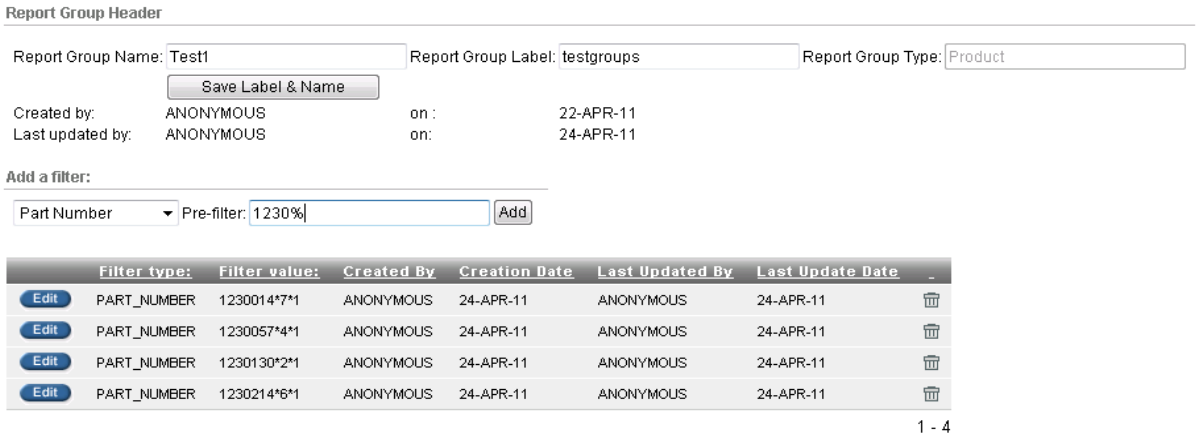


12. ábra

Erre azért van szükség, mert a range és a filter **alap** típusú groupok nagyban különböznek. A range típusúnál csak egy alsó és egy felső érték megadására van lehetőség a group típusának megfelelően, jelenleg csak a postal code típusnál, ahol egy irányítószám tartományt tudunk megadni.
A filter alap típusú groupoknál pedig az adott grouptípusnál rendelkezésre álló filterekből tetszőleges számút adhatunk hozzá, ahol minden filter egy értéket definiálhat.

5.4 Report Group módosítása oldal

Ezen az oldalon van lehetőség egy újonnan létrehozott vagy már meglévő group részleteinek megadására. Hozzáadhatunk a filter választékból vagy módosíthatunk egy meglévő hozzáadott filtert, vagy pedig ki törölhetjük azt az éppen szerkesztett groupból. A követelményeknél megfogalmaztuk, hogy melyik group milyen filtert tartalmazhat. Minden filterre íratlan szabályként megadtuk, hogy töltse ki a user a Pre-filter input mezőt.



13. ábra

A pre-filtering azért került bele az alkalmazásba, mert amikor egy filtert akarunk hozzáadni egy grouphoz, akkor a felhasználó választhat egyes esetekben hatalmas mennyiségű adatot tartalmazó listából több elemet is.

Ez komoly teljesítménybeli problémákat okoz az APEX számára, hiszen adatbázis linken keresztül kell áthoznia az adatokat.

Tehát egy filter egy adott tábla egy oszlopának (attribútumának) értékei közül lehet kiválasztva, vagyis az alkalmazás ekkor egy adatbázis linken keresztül egy ilyen formájú lekérdezés eredményeit teszi elérhetővé:

```
SELECT distinct  part_number
  FROM nidw_product_categories
WHERE part_number LIKE prefilter_ertek
MINUS
SELECT g_value part_number
  FROM nidw_level_rpt_groups
 WHERE DW_RPT_GROUP_ID =1
    AND g_level ='PART_NUMBER';
```

1. kódrészlet

A lekérés két részre bontható a MINUS körül, az első SQL felhozza, ebben az esetben a termék azonosítókat az azonosakat kiszűrve (distinct part_number), a másik lekérés pedig az adott report grouphoz tartozó (itt dw_rpt_group_id = 1), azonos típusú filtereket szelektálja fel.

Az így kapott lekérdezés tartalmazni fogja a PRE_FILTER értékkel leszűrt, összes termék azonosítót a már felhasználtak nélkül.

Természetesen itt a LIKE után a prefilter_ertek egy ilyen stringet jelent: ”LIKE ’1234%’”, de ennek a generálására a későbbiekben részletesebben ki fogok térni.

A „Save Label & Name” gombbal elmenthetjük a megváltoztatott név és címke értékeket, ha akarjuk.

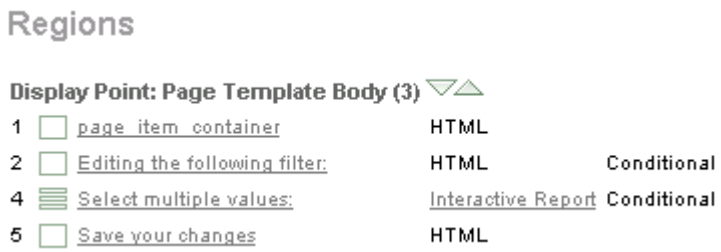
Az ábrán alul elhelyezkedő interaktív riport tartalmazza a már hozzáadott filtereket. Itt is láthatjuk, hogy ki és mikor hozta létre, vagy mikor lett utoljára módosítva és ki által. Törölhetjük őket vagy megváltoztathatjuk, ha kívánjuk.

Ha kiválasztjuk a hozzáadni kívánt filter típust, majd megadjuk az előszűrési feltételeket a pre-filter mezőben, akkor a „Filterek módosítása” oldalra kerülünk, de akkor is, ha már egy

meglévő filtert kívánunk módosítani az interaktív riport balszélső oszlopában az EDIT gombra kattintva.

5.5 Filterek módosítása oldal

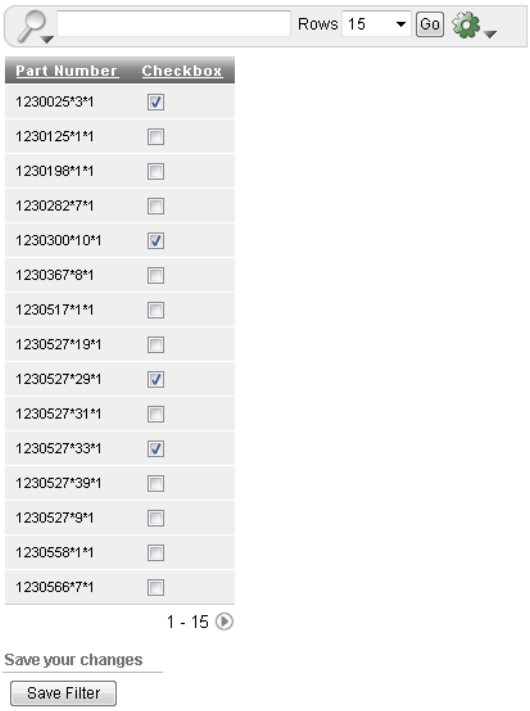
Ezen az oldalon adhatunk hozzá vagy módosíthatunk egy meglévő filtert. Az oldal viselkedése attól függ, hogy egy P4_IS_SAVING nevű változó értékét miszerint állítjuk be az előző oldalról ide navigálva.



14. ábra

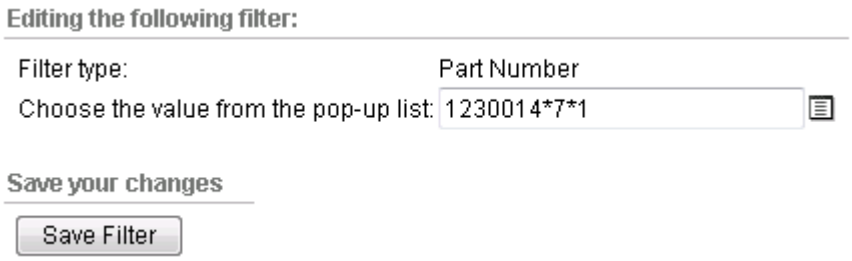
Ha egy új filtert akarunk hozzáadni, akkor a P4_IS_SAVING értéke ”FALSE” lesz, vagyis megkapja a user a teljes listát a kiválasztható értékekről. Ez úgy történik, hogy a „Select multiple values:” nevű interaktív riport megjelenítése feltételhez van kötve, a feltétel a következő:

Feltétel típusa: (Az első kifejezésben szereplő elem értéke) **!=** (a második kifejezéssel)
1.Kifejezés: (P4_IS_SAVING) **2.Kifejezés:** (TRUE)



15. ábra

Ha a P4_IS_SAVING nincs megadva, vagy TRUE, akkor pedig az „editing the following filter” HTML régió jelenik meg ahol látható a filter típusa, egy „Popup LOV” és egy mentés gomb.

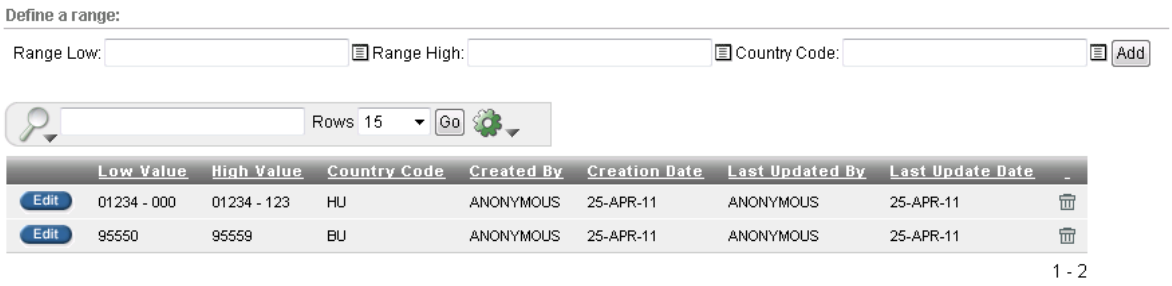


16. ábra

A Popup LOV (Popup List Of Values) szintén egy nagyszerű APEX által kínált eszköz, az input mezőben megadható egy érték és a mező jobb szélén látható kis ikonra kattintva a mezőben megadott értékekhez hasonló értékekből fog feljönni egy lista, ahonnan a user kiválaszthatja a megfelelő hozzáadni kívánt értéket vagy egy másik keresést hajthat végre az értékek között.

Az interaktív riportnál implementáltam egy nagyon érdekes és hasznos megoldást az elemek kiválasztására és az interaktív riport mögötti lekérések dinamikus generálására. Erre is ki fogok térni az előző lekérdezésekkel együtt, amelyek visszaadták a már kiválasztott értékek nélkül a még választható értékek halmazát.

5.6 Range Type Group módosítása oldal



17. ábra

Az oldalon három input mező található. A ”Range Low” a postai irányítószámok intervallumának alsó határa, a ”Range High” pedig a felső határa, ezek mellett még meg kell adni a ”Country Code” mezőbe is egy rövidített álló ország azonosítót, de ha a Popup Lov ikonjára kattintunk, akkor kiválaszthatjuk az ország teljes neve alapján.

Hozzáadhatunk új intervallumokat, törölhetünk belőlük vagy módosíthatjuk őket. Külön oldal kellett rá, mivel a működési elve, ahogyan már említettem más, mint a filtereké.

6.Problémák és megoldásaik

6.1.1 Probléma: 20 különböző LOV megadása, kezelhetősége

A LOV-ok(List Of Values) nagy szerepet játszanak az alkalmazásban, ezekkel tudunk értékeket kiválasztani már meglévő filtereknél vagy meglévő,új intervallumoknál.

Egy LOV mögött egy SQL lekérdezés áll, ami visszaadja egy adott tábla egy oszlopában lévő értékeket. (lásd 1. kódrészlet).

Viszont ha több LOV-t szeretnénk használni, akkor ezek mind-mind külön álló Popup Lov elemek lesznek az oldalon. Az alkalmazás esetében 20 darab LOV-t kellett volna egyesével elkészíteni. Ez azt jelentette volna, hogy az alábbi elemek létrehozását 20-szor kellett volna végig csinálni:

Elemek:

1. régió az elemekhez, feltétel megadása, hogy mikor jelenhet meg, vagyis, hogy az adott típusú filterhez megfelelő elemeket akarjuk lekérdezni.
2. Popup Lov mező hozzáadása.
3. Az SQL lekérdezés megírása.

Ez egy repetitív/ismétlődő feladat és az SQL kód változtatásához nem megfelelő megoldás, mind emellett, ha új filter típus kívánunk hozzáadni az alkalmazáshoz, ezt újból végig kellene csinálni.

6.1.2 A megoldás: Dinamikus LOV-ok

A megoldás lényege abban áll, hogy egy LOV forrásában nem csak SQL lekérdezést adhatunk meg, hanem akár egy PL/SQL blokkot is.

A group típusból és a filter típusból eldönthetjük, hogy milyen adatokra van szükségünk, milyen SQL-t kell lefuttatnunk, hogy megkaphassuk őket.

Legyen két változó, amely tárolja, hogy milyen group típusról és milyen filter típusról vagy range típusról van szó.

Készítsünk egy PL/SQL blokkot, ahol feltételekkel vizsgáljuk a két változó értékét.

Ha egy feltétel teljesül, adjuk vissza a kívánt lekérdezést.

A PL/SQL kódból egy részlet:

```
BEGIN
IF :P8_RPRT_GTYPE = 'Postal Code'
THEN RETURN
'select distinct postal_code d,postal_code r from nidw_locations ORDER BY 1';
ELSIF :P4_RPRT_FTYPE = 'BOOKINGS_CLASS'
THEN RETURN
'select distinct bookings_class d, bookings_class r from nidw_product_categories order by 1';
ELSIF :P4_RPRT_FTYPE = 'PRODUCT_FAMILY'
THEN RETURN
'select distinct product_family d, product_family r from nidw_product_categories order by 1';
.
.
.
END IF;
END;
```

2. kódrészlet

A ”d” és ”r” aliasok azért vannak, mert egy LOV-nál megadhatjuk a megjeleníteni kívánt display értéket és a hozzá tartozó return értéket.

Például: **display value**: Magyarország **return value**: HU

Ezzel a megoldással egy helyen kezelhetjük a lekérdezéseket, könnyen módosíthatjuk vagy bővíthetjük őket. Nem kell 20 külön álló Popup LOV-ot létrehozni, egyesével manuálisan. Rengeteg problémát elkerülhetünk vele, köztük azt, hogy 20-féle megjelenítési feltételt is meg kellene adnunk az elemekhez. Így egy régió lett hozzáadva, egy feltétel, mégpedig az, hogy éppen meg akarunk változtatni egy már hozzáadott filtert vagy intervallumot.

6.2.1 Probléma: 20 külön álló interaktív riport megadása és ezeken több elem egyszerre történő kiválasztása

Az interaktív riportokat nem csak a meglévő filterek és meglévő groupok megjelenítésére használom az alkalmazásban, hanem az értékek kiválasztásánál is a filter típusnak megfelelően. Ez a probléma nem ugyanaz, mint az előző, mert azt csak a módosításnál használtam, vagyis csak egy érték megadására van lehetőség.

Mivel a végfelhasználók jelezték, hogy nem csupán egyesével szeretnék az értékeket hozzáadni, mert az nagyon időigényes például 50-100 darabos filter számnál, ezért ki kellett találni valami megoldást.

6.2.2 A megoldás: Dinamikus Interaktív Riport + Checkbox + Javascript + Application Process

A lekérdezéseket szintén dinamikusan kapjuk meg egy PL/SQL csomag egy függvényétől a filter típusnak, és group típusnak megfelelően. A csomag az adatpiac adatbázisán van.

```
CREATE OR REPLACE PACKAGE BODY report_groups_utils
AS
/*****************************************************************
Name: get_query_for_collection
Description:  Returns a sql query for the given parameters for use in the apex_collection for
report groups
Created by: Vilmos Katona
*****************************************************************/
FUNCTION get_query_for_collection(rpt_filter_type_p IN VARCHAR2,
                                  rpt_group_type_p IN VARCHAR2...
```

3. kódrészlet

Ezzel a megoldással szintén csak egy interaktív riportra van szükségünk.

Az első problémát ezzel megoldottam, nem kell 20 darab külön riportot létrehozni, elég csak egyet, de továbbra is fenn áll az, hogy miként lehet elérni, hogy több értéket is tudjunk kiválasztani, továbbá megjelenik a probléma, hogy a felhasználó lehet, hogy nem csupán az első oldalról akarja kiválasztani, hanem lapozni akar az interaktív riportban és folytatni a kijelöléseket.

Erre megoldás a checkboxok használata, amivel több értéket is meg tudunk jelölni. A kijelölt értékek elmentésére egy application processen keresztül van lehetőség, amit javascriptel fogok meghívni.

Az **application process** egy PL/SQL blokk formájában jelenik meg apexben. A megosztott objektumok(Shared Components) között tudjuk definiálni. Két fő típusa van: az egyik minden alkalmazás oldalon lefut egy meghatározott pillanatban, a másik pedig csak on-demand módon, vagyis ha egy kérés érkezik a process felé.

Én az utóbbit használtam fel arra, hogy a kijelölt elemeket az interaktív riportból eltároljam.

Tehát a user kiválasztja az adott filter típust, megjelenik számára egy interaktív riport a pre-filtering feltételeknek megfelelően, ahol két oszlopot lát. Az egyik az értékek, amiket kiválaszthat a másik pedig az értékekhez tartozó checkbox mezők, amit be tud pipálni, ha az értéket hozzá akarja adni.

Minden kattintásnál a checkboxon egy javascript onclick esemény váltódik ki.

```
<input type="checkbox" name="f01" value="1234061*1*1" onclick="checkchange(this);" />
```

4. kódrészlet

Ekkor meghívódik az application process, ami egy változóban ":"-al delimitáltan eltárolja a kiválasztott értékeket.

Ha kivesszük a checkbox bejelölését, akkor az adott érték törlődik a listából.

Az interaktív riport forrása nem egy közvetlen lekérdezés lesz az adatbázis felé, hanem egy APEX kollekció. Az APEX kollekció segítségével tudjuk megoldani azt, hogy ugyanaz az interaktív riport mögött különböző SQL lekérdezések lehessenek.

Ahogy betölt az oldal, lefut egy page level process, vagyis egy olyan folyamat, ami egy adott belépési ponton indul el. Ez a process point-ban definiált On Load – Before Header, ami azt jelenti, hogy még a header tartalom elküldése előtt lefut.

Ez a process a P4_RETURN_IRQ nevet kapta, ami a “Return Interactive Query”-ből jön. A folyamat egy PL/SQL-ben írt feltétel. Ha létezik már kollekció, akkor azt kitöröljük, eldobjuk, hiszen ez hatalmas memória-tár erőforrásokat vesz igénybe, nem csinálhatunk többet egymás mellett, meg egyébként is egy ilyen objektumot használunk minden filterre.

Ezek után létrehozzuk a kollekciót, a dinamikus generált SQL lekérdezés alapján.

```
IF ( APEX_COLLECTION.COLLECTION_EXISTS ('DYNAMIC_IR') ) THEN
BEGIN
apex_collection.delete_collection(p_collection_name => 'DYNAMIC_IR');
APEX_COLLECTION.CREATE_COLLECTION_FROM_QUERY_B(
p_collection_name => 'DYNAMIC_IR',
p_query => report_groups_utils.get_query_for_collection
(:P4_RPRT_FTYPE,:P4_RPRT_GTYPE,:P4_RPRTGIDHOLDER,:P4_RPRT_PREFILTER) );
END;
ELSE APEX_COLLECTION.CREATE_COLLECTION_FROM_QUERY_B(
p_collection_name => 'DYNAMIC_IR',
p_query => report_groups_utils.get_query_for_collection
(:P4_RPRT_FTYPE,:P4_RPRT_GTYPE,:P4_RPRTGIDHOLDER,:P4_RPRT_PREFILTER) );
END IF;
```

5. kódrészlet

Létrejött a kollekció, ekkor az interaktív riport forrása a következő lesz.

```
SELECT apex_item.checkbox (1,
                    C001,
                    'onclick="spcheckchange(this);"',
                    :f_partno_list,
                    ':'
                   ) checkbox,C001
FROM apex_collections
WHERE collection_name = 'DYNAMIC_IR';
```

6. kódrészlet

Ez a forrás mindig így néz ki, nem ez változik, hanem az APEX kollekció mögötti SQL lekérdezés.

Ahhoz, hogy megértsük, miért maradnak meg a kijelölések az interaktív riportnál, meg kell vizsgálnunk az apex_item.chekbox függvényt.

Ez a függvény checkboxokat hoz létre, az adott sorokhoz. Öt paramétere van.

```
APEX_ITEM.CHECKBOX(
    p_idx                       IN    NUMBER,
    p_value                     IN    VARCHAR2 DEFAULT,
    p_attributes                IN    VARCHAR2 DEFAULT,
    p_checked_values            IN    VARCHAR2 DEFAULT,
    p_checked_values_delimiter  IN    VARCHAR2 DEFAULT)
    RETURN VARCHAR2;
```

Az első parameter a p_idx, amely meghatározza, hogy melyik globális változót fogja használni.

A második parameter a p_value, amely meghatározza a checkbox értékét.

A harmadik a p_attributes paraméterrel határozhatjuk meg a checkboxhoz hozzáadható plussz attribútumokat.

A negyedik parameter a p_checked_values, mely arra való, hogy a már kijelölt értékeket tartalmazó változó nevét adhassuk meg benne, ami egy string.

Az ötödik parameter a p_checked_values_delimiter pedig arra, hogy az értékeket elválasztó karaktert határozhassuk meg vele.

Így az interaktív riportunkból csak egyet kellett létrehoznunk, aminek a forrása egy APEX kollekció, ami mögött bármilyen SQL lekérdezés állhat. Az értékek mellé checkboxokat hoztunk létre, amivel a user több értéket is megjelölhet, s a lapozásnál nem tűnnek el a kijelölt értékek, mert azok egy globális változóban tárolódnak. Egy dologra kell még figyelnünk, hogy a művelet után ürítsük ki a globális változót, ne maradjon benne adat, mert ez az oldal újratöltésénél gondokat okozhat.

6.3 Az alkalmazás fejlesztés hátterének áttekintése

A Report Groups fejlesztése, ahogyan minden egyéb APEX alkalmazás a cégnél, fejlesztői rendszeren kezdődik. A jogosultságok itt sokkal bővebbek, hogy minél könnyebben oldhassuk meg a problémákat. Ha kiderül valamilyen hiányosság adatbázis szinten vagy a környezettel kapcsolatban, akkor a fejlesztők szabadon módosíthatják azt.

Ahogy az alkalmazás elkészült, a fejlesztői rendszeren el kell készíteni egy install szkriptet. Az install szkript feladata, hogy beimportálja az alkalmazást, létrehozza a táblákat, szinonimákat, csomagot, mindent, ami az alkalmazás működéséhez szükséges lehet.

Ezen a rendszeren már nem tudunk manuálisan jogosultságokat változtatni vagy kézzel létrehozni objektumokat. Ez a rendszer már sokkal zártabb, pont azért, hogy kiderüljenek a problémák még az éles rendszerre történő mozgatás előtt. Ha nem sikerült az alkalmazásunkat feltelepíteni a teszt környezetre, akkor az install szkript naplófájljából kideríthetjük, hogy mi lehetett a gond.

A tesztelés megkezdődhet, s azt alaposan és körültekintően kell végezni. Az Üzleti Elemző feladata az Elemző Programozó mellett, hogy tesztelje az elkészült alkalmazást, és ha hibákat vagy nem megfelelő viselkedés módokat talál, akkor a fejlesztő felé továbbítsa azokat miden részletükkel együtt (mikor, milyen körülmények között jött elő, hogyan reprodukálható).

Amint a tesztelés végére értünk és az alkalmazásunk kész az éles rendszerre való kihelyezéshez, egyeztetni kell az adatbázis adminisztrátorokkal, hogy mikor és kinek a felügyelete alatt történhet meg a telepítés.

A telepítés befejeztével validálni kell a kihelyezett alkalmazást, hogy meg van e minden funkcionalitás, nem léptek-e fel problémák a telepítés közben, minden rendben van-e.

Az Üzleti Elemző is újból leteszteli a rendszert, kipróbálja a funkciókat, és jelzi, ha bármilyen probléma fellépett volna.

Ha egy új verzió fejlesztésébe kezdenénk bele, akkor a folyamat újból a fejlesztői környezeten kezdődne, folytatván az utolsó verzióját az alkalmazásnak, s a lépések ugyanezek maradnának.

Újból megkezdődne az igények felmérése, a követelmény rendszer felállítása, a követelmények meghatározása.

Ha az alkalmazás elérné azt a kort vagy státuszt, amikor már nem érdemes vagy túl költséges annak tovább fejlesztése, azt működésen kívül helyeznék, amint meg van a feladatokat ellátó új rendszer, s az bizonyítottan kiválthatja a régit.

6.4 Trainingek

A cégnél, ha egy programozó számára eddig ismeretlen vagy kevésbé ismert technológiával, eszközzel kapcsolatban kap feladatot, trainingeken kell részt vennie. Ezt az adott témában úgymond expert vagy tapasztaltabb fejlesztő tarthatja.

A trainingek több alkalommal, négy vagy több szem közt zajlanak. Megadnak minden szükséges információt az elinduláshoz és mindig biztosítanak lehetőséget a kérdezésre, s ha szükséges, a dolgok újbóli vagy bővebb kifejtésére.

A gyakorlottabb fejlesztők által tartott trainingeken mellett rengeteg dokumentáció, audiovizuális anyag érhető el a fejlesztő számára, s emellett bármilyen szükséges nyomtatott kiadványhoz hozzáférhet.

Kiemelném, és számomra nagyon sok segítséget és támogatást nyújtott, hogy a tapasztaltabb vagy akár expert fejlesztők is bármilyen kérdésemre válaszoltak, vagy egy probléma megoldására közös brainstormingot tettek lehetővé. Így nagyon fontos szerepet kap a kommunikáció nálunk, s az emberek közötti együttműködés szinte nem engedi, hogy leragadjunk egy adott problémánál.

6.5 SDLC az NI-nál (Software Development Lifecycle)

1. Request Management(Problémakezelés)

- A problémákat és az új fejlesztési igényeket követjük, és felülvizsgáljuk, mielőtt egy szoftverkiadási verzióba belekerülnének.

2. Project Formation (Projekttervezés)

- Egy projekt tervezése a felülvizsgált problémák halmazából.

3. Analysis

- A problémák elemzése a megvalósíthatóság és a lehetséges megoldások szempontjaiból. Ez a lépés magába foglalja az igények összegyűjtését és követelmények elfogadását.

4. Design (Tervezés)

- A megoldás kidolgozása, majd elkészülte után annak felülvizsgálása.

5. Build (Implementáció)

- A project követelményeinek megvalósítása, majd az Üzleti oldal által történő tesztelésre való felkészülés.

6. Test (Tesztelés)

- Az elkészült szoftver tesztelése, a problémák azonosítása, megoldása. Training anyagok és dokumentációk elkészítése.

7. Release (Szoftver kiadás)

Az elkészült termék élesbe állítása, felsetupolása. A végfelhasználók kiképzése annak használatára.

7.Összefoglalás

A szoftver elkészült és folyamatosan elérhető az éles rendszeren. Nem sikerült teljes mértékben minden, a követelményekben foglalt részkövetelményt teljesíteni. Ennek az oka egyrészt az idő szűke, mivel egy új magasabb prioritású projektben kell részt vennem.

Egy ilyen rész például az authentikáció, amit az LDAP csoportokkal kívántunk megvalósítani. Ahogy közeledett a határidő, úgy változott meg az eszköz hozzáférhetőségének definiálása. Eleinte azzal számoltunk, hogy csak egy bizonyos felhasználói halmaz férhet hozzá az alkalmazáshoz, de a későbbiekben az üzleti oldalról az a kijelentés jött, hogy ne legyen lekorlátozva, bárki beléphessen és használhassa azt.

Ez azzal a látható eredménnyel járt, hogy mivel elmarad a szoftver indításának elején az authentikáció, nem lehet követni, hogy ki, milyen módosításokat vitt véghez, s ezt az okozza, hogy a változó, amely a beléptetett felhasználó nevét tartalmazza, nem kap értéket, minden felhasználási helyén az ANONYMOUS érték áll.

A későbbiekben fogok elkészíteni, egy új verziót, ahol lesz egy olyan LDAP group, ami a vállalat minden felhasználóját fogja tartalmazni, és így ez a hiba meg fog szűnni. Feladatom volt továbbá, hogy az alkalmazás működését és a megoldásaimat el kelljen magyarázni más fejlesztőnek is, márpedig azért, hogy ha én valamilyen egyéb projekten dolgozok, de szükség van egy magas prioritású probléma kijavítására, vagy valamilyen rendkívüli üzleti igény alkalmazás szintű kielégítésére, ne kelljen azzal várni, amíg be nem fejeztem a másik munkám.

Ez training formájában történt meg, ahol ismertettem az alkalmazást, és annak fontos részeit, ismert hibáit, vagy bármilyen fejlesztési nehézséget, amivel találkoztam és érdemes rá figyelni.

A továbbiakban már készülőben van egy dokumentum, amely az új észrevett apróbb problémákat, vagy újonnan felmért követelményeket fogja tartalmazni. Ennek a dokumentumnak a kezeléséről a projekthez tartozó Üzleti Elemző felel. Ő az, aki követi a problémákat, és fenntartja a kommunikációt a túloldallal.

Az APEX nem egy kis eszköz, rengeteg apró részből áll össze ugyanúgy, mint bármilyen más hasonló felhasználási területű fejlesztői eszköz. Alapvetően, ami nehézségeket okozott eleinte az, hogy semmilyen megelőző tapasztalatom nem volt vele. Rengeteget segítettek a bevezető trainingek, ahol megismerkedhetem az alapvetően ismeretekkel, majd a

későbbiekben, mind a prezentációk formájában elérhető trainingek, mind a munkatársaktól kapott segítségek.

Újból hangsúlyoznom kell, hogy nagyon fontos volt a fejlesztés folyamán a kommunikáció. Az amerikai managerem például minden héten várt tőlem egy “status reportot”, egyfajta helyzetjelentést, amiben meg kellett fogalmaznom az aktuálisan véget érő héten elért céljaimat és a terveimet a következő hétre.

Meg tanultam jobban tervezni a feladatokat, jobban becsülni a befejezésükhöz párosítható munkaidőt. Elválasztani a háttér információk megszerzésére szánt időt és az előbb említett valóban az implementálásra fordított időt.

Nem ez volt az első önálló projektem, de a tervezés fontossága és a pontos követelmények megfogalmazásának szüksége szempontjából nagyon sokat tanultam belőle.

Fejlődtem a komplex problémák átlátásának képességében és azok megközelítésének helyes módjának kiválasztásában.

A fejlesztési idő két hónap volt, az nyolc hét, heti 25 órában. Az első 2-3 hét nehézségei után és az eszköz jobb megismerését követően, sokkal többet tudtám már az alkalmazás logikájának megvalósítására szánni, nem kellett a különböző apróságok keresésével és megértésével foglalkoznom.

Az eszközről kialakult véleményem vegyes, az APEX nagyon sok szempontból valóban kézre álló, tényleg rengeteg beépített modult, varázslót és építődarabot ad a fejlesztő kezébe, amivel adott problémák könnyen és nagyszerűen megoldható.

Viszont sokszor pont emiatt, hogy előre elkészített elemeket tartalmaz, nem egyszerű ezeket az adott részegységeket testre szabni, megváltoztatni. Kiváltani pedig szinte teljesen lehetetlen vagy nagyon sok munkát és körülményeskedést kívánna. Ami valóban tetszett és mondhatni örömömet leltem, az a PL/SQL nyelv sok helyen történő felhasználhatósága.

Figyelemmel fogom követni az APEX új verzióinak megjelenését és az új lehetőségeket, amiket nyújtani fognak, mert bármilyen új probléma vagy igény merül fel a report groupsal kapcsolatban, lehet egyszerűbb vagy hatékonyabb megoldást találok rá, ezáltal megnövelve élettartalmát és könnyítve mind a végfelhasználók és mind az alkalmazás későbbi fejlesztőinek munkáját.

Látni a rendszert működni, és tudni azt, hogy hasznos, amit alkottam nagyszerű érzés.

“Information is beautiful”

8.Irodalomjegyzék

Könyvek:

[1]Packt Publishing - Oracle Application Express 3.2

Arie Geller; Matthew Lyon

[2]Apress - Beginning Oracle Application Express 4

Doug Gault; Karen Cannell; Patrick Cimolini; Martin D'Souza; Timothy St. Hilaire

Apex online dokumentációk:

[3]http://www.oracle.com/technetwork/developer-tools/apex/application-express/apex-094287.html

OBIEE online dokumentációk:

[4]http://www.oracle.com/us/solutions/ent-performance-bi/enterprise-edition-066546.html

Training anyagok:

Apollo:

[5]Apollo Introduction

[6]Apollo Basics

Apex:

[7]Lesson 1 - Creating an Application
[8]Lesson 2 - Creating a Report
[9]Lesson 3 - Adding Pages and Interactive Reports
[10]Lesson 4 - Creating Forms
[11]Lesson 5 - Integrating with NI Apex AA and the Menu App
[12]Lesson 6 - Deploying your application
[13]Lesson 7 - Modifying Application Themes
[14] Követelményeket tartalmazó dokumentum (Report Groups Requirements)

9.Függelék

Business Value for Report Groups

You have delivered significant business value when you create a tool that will be leveraged by company vice president's in order to get data in order to make decisions when those decisions need to be made rather than being forced to wait on data requests to Business Intelligence or Information Technology groups. This tool will allow that data to be self service for these individuals and get their data when they need it.

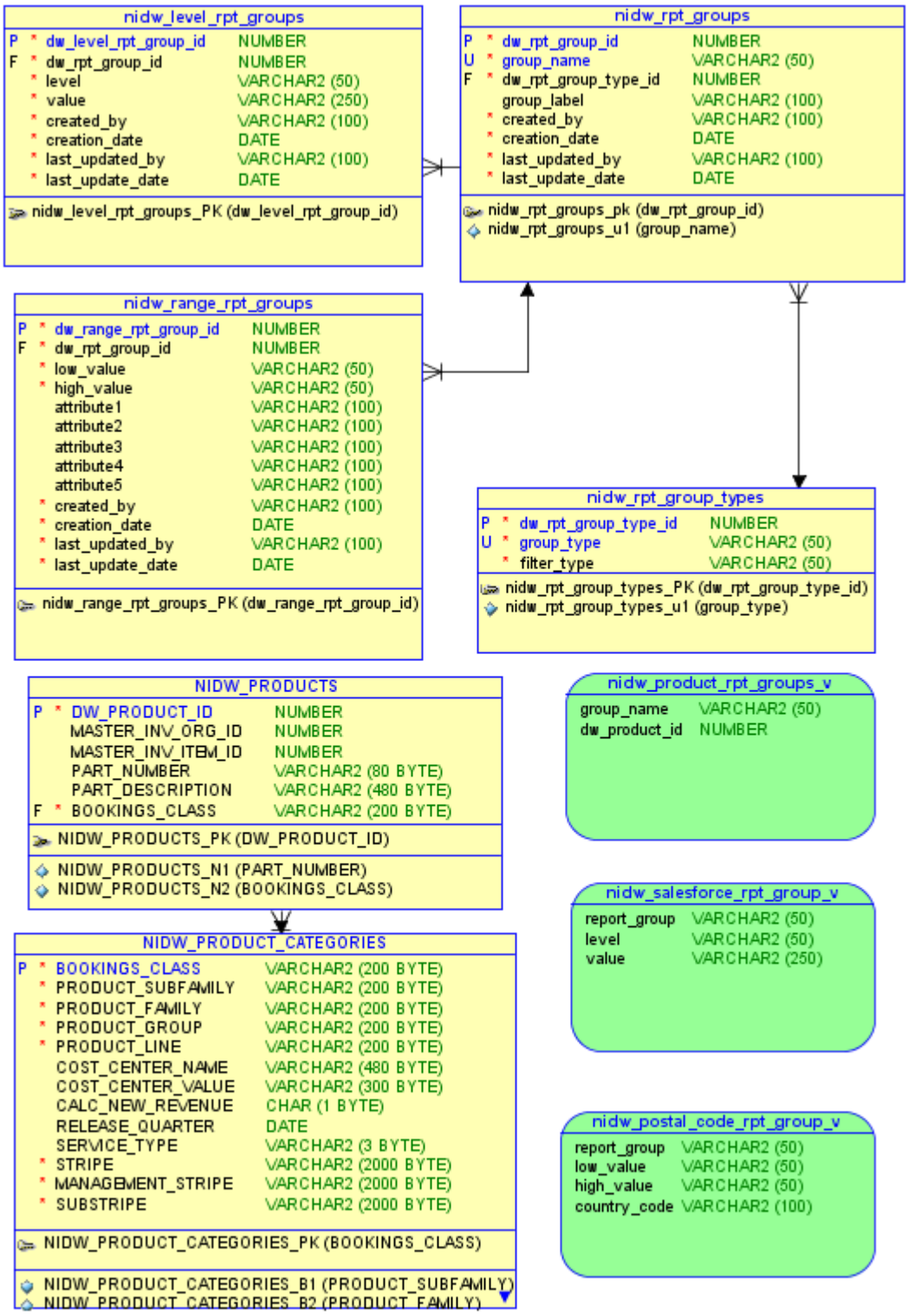
This Report Groups Tool integrates very smoothly with Oracle Business Intelligence Enterprise Edition. It extends the out of the box functionality to allow for more dynamic Dashboards and Reports that scale to meet the ever changing business needs at National Instruments.

First tool that we've had at National Instruments to enable multiple End User Reporting solutions to leverage customized data sets. This represents a large amount of efficiency for complex analytics Business Intelligence Users who must use multiple tools for Reporting, they no longer spend duplicate time maintaining these values in multiple places.

Vilmos really created a very smooth End User Interface. There was really very little need to create detailed documentation for End Users use of the tool, when you open the application it is very intuitive in it's use.

Brent Freeman

Datamodel


nidw_level_rpt_groups
P * dw_level_rpt_group_id NUMBER
F * dw_rpt_group_id NUMBER
* level VARCHAR2 (50)
* value VARCHAR2 (250)
* created_by VARCHAR2 (100)
* creation_date DATE
* last_updated_by VARCHAR2 (100)
* last_update_date DATE
nidw_level_rpt_groups_PK (dw_level_rpt_group_id)
nidw_rpt_groups
P * dw_rpt_group_id NUMBER
U * group_name VARCHAR2 (50)
F * dw_rpt_group_type_id NUMBER
group_label VARCHAR2 (100)
* created_by VARCHAR2 (100)
* creation_date DATE
* last_updated_by VARCHAR2 (100)
* last_update_date DATE
nidw_rpt_groups_pk (dw_rpt_group_id)
nidw_rpt_groups_u1 (group_name)
nidw_range_rpt_groups
P * dw_range_rpt_group_id NUMBER
F * dw_rpt_group_id NUMBER
* low_value VARCHAR2 (50)
* high_value VARCHAR2 (50)
attribute1 VARCHAR2 (100)
attribute2 VARCHAR2 (100)
attribute3 VARCHAR2 (100)
attribute4 VARCHAR2 (100)
attribute5 VARCHAR2 (100)
* created_by VARCHAR2 (100)
* creation_date DATE
* last_updated_by VARCHAR2 (100)
* last_update_date DATE
nidw_range_rpt_groups_PK (dw_range_rpt_group_id)
nidw_rpt_group_types
P * dw_rpt_group_type_id NUMBER
U * group_type VARCHAR2 (50)
* filter_type VARCHAR2 (50)
nidw_rpt_group_types_PK (dw_rpt_group_type_id)
nidw_rpt_group_types_u1 (group_type)
NIDW_PRODUCTS
P * DW_PRODUCT_ID NUMBER
MASTER_INV_ORG_ID NUMBER
MASTER_INV_ITEM_ID NUMBER
PART_NUMBER VARCHAR2 (80 BYTE)
PART_DESCRIPTION VARCHAR2 (480 BYTE)
F * BOOKINGS_CLASS VARCHAR2 (200 BYTE)
NIDW_PRODUCTS_PK (DW_PRODUCT_ID)
NIDW_PRODUCTS_N1 (PART_NUMBER)
NIDW_PRODUCTS_N2 (BOOKINGS_CLASS)
nidw_product_rpt_groups_v
group_name VARCHAR2 (50)
dw_product_id NUMBER
nidw_salesforce_rpt_group_v
report_group VARCHAR2 (50)
level VARCHAR2 (50)
value VARCHAR2 (250)
NIDW_PRODUCT_CATEGORIES
P * BOOKINGS_CLASS VARCHAR2 (200 BYTE)
* PRODUCT_SUBFAMILY VARCHAR2 (200 BYTE)
* PRODUCT_FAMILY VARCHAR2 (200 BYTE)
* PRODUCT_GROUP VARCHAR2 (200 BYTE)
* PRODUCT_LINE VARCHAR2 (200 BYTE)
COST_CENTER_NAME VARCHAR2 (480 BYTE)
COST_CENTER_VALUE VARCHAR2 (300 BYTE)
CALC_NEW_REVENUE CHAR (1 BYTE)
RELEASE_QUARTER DATE
SERVICE_TYPE VARCHAR2 (3 BYTE)
* STRIPE VARCHAR2 (2000 BYTE)
* MANAGEMENT_STRIPE VARCHAR2 (2000 BYTE)
* SUBSTRIPE VARCHAR2 (2000 BYTE)
NIDW_PRODUCT_CATEGORIES_PK (BOOKINGS_CLASS)
NIDW_PRODUCT_CATEGORIES_B1 (PRODUCT_SUBFAMILY)
NIDW_PRODUCT_CATEGORIES_B2 (PRODUCT_FAMILY)
nidw_postal_code_rpt_group_v
report_group VARCHAR2 (50)
low_value VARCHAR2 (50)
high_value VARCHAR2 (50)
country_code VARCHAR2 (100)

10. Köszönetnyilvánítás

Szeretném megköszönni a szakdolgozat megírásához kapott segítséget **Kollár Lajosnak** a témavezetőmnek, a türelmét és a tanácsait, építő meglátásait.

Vágó Csabának a külső témavezetőmnek, a National Instruments Hungary Kft. Datawarehouse and Finance menedzserének, akinek csapatban gyakornokként dolgozok. Bátorításáért, a projektért (vagyis, hogy megkaphattam azt), és segítségeiért.

Rushika Pandyanak az Austini menedzseremnek, a projekt felügyeletéért, a heti beszámolóim követéséért és a felém támasztott elvárásaiért.

Brent Freemannek a projektért felelős Üzleti Elemzőnek és **Greg Duffeynak** a támogató architektnek, valamint **Németh Márknak** az adattárház csapatunk Team Leadjének a tervezésben való hatalmas segítségért és a folyamatok megismertetéséért.

Vas Dávid elemző programozónak az APEX megismerési folyamatának elején nyújtott segítségeiért.

Komlósi Lászlónak a segítségért, amikor már a fáradtságtól alig láttam, és megcsinálta nekem a csomagokat az alkalmazás éles rendszerre történő kihelyezéséhez, amíg én az utolsó simításokat végeztem.

Az adattárházas csapat eddig nem meg említett tagjainak a türelméért, és bátorításáért névszerint: **Thurzó Ákos,Löki Levente,Vukovich László, Bónizs Attila, Hennel Zoltán, Gargya Attila**

A családomnak és barátaimnak, hogy elviseltek a nehéz pillanataimban és támogattak.

Köszönöm.

11.Plágium - Nyilatkozat

Szakdolgozat készítésére vonatkozó szabályok betartásáról nyilatkozat

Alulírott (Neptun-kód: …NXMUL5…) jelen nyilatkozat aláírásával kijelentem, hogy az

Alkalmazás fejlesztés Oracle eszközökkel

című szakdolgozat/diplomamunka

(a továbbiakban: dolgozat) önálló munkám, a dolgozat készítése során betartottam a szerzői jogról szóló 1999. évi LXXVI. tv. szabályait, valamint az egyetem által előírt, a dolgozat készítésére vonatkozó szabályokat, különösen a hivatkozások és idézések tekintetében.

Kijelentem továbbá, hogy a dolgozat készítése során az önálló munka kitétel tekintetében a konzulenst, illetve a feladatot kiadó oktatót nem tévesztettem meg.

Jelen nyilatkozat aláírásával tudomásul veszem, hogy amennyiben bizonyítható, hogy a dolgozatot nem magam készítettem vagy a dolgozattal kapcsolatban szerzői jogsértés ténye merül fel, a Debreceni Egyetem megtagadja a dolgozat befogadását és ellenem fegyelmi eljárást indíthat.

A dolgozat befogadásának megtagadása és a fegyelmi eljárás indítása nem érinti a szerzői jogsértés miatti egyéb (polgári jogi, szabálysértési jogi, büntetőjogi) jogkövetkezményeket.

Katona Vilmos hallgató

Debrecen, 2011.04.30.