Eiropas Reģionālās attīstības fonds

Prioritāte: 2.1. Zinātne un inovācijas

Pasākums: 2.1.1. Zinātne, pētniecība un attīstība

Aktivitāte: 2.1.1.1. Atbalsts zinātnei un pētniecībai

Projekts: ”Multi - modeļu izstrādes tehnoloģija .NET pielietojumu projektiem”

Projekta sākuma datums: 2014.gada 1.janvāris.

Projekta beigu datums: 2015.gada 30.jūnijs.

Līguma Nr. 2013/0031/2DP/2.1.1.1.0/13/APIA/VIAA/010

ESF finansējuma saņēmējs: SIA, SWH SETS

Sadarbības partneris: Elektronikas un datorzinātņu institūts (EDI)

Projekta aktivitātes Nr.3.3 "Datu bāzes meta modeļa izstrāde un pētniecība, kas ietver dažādu datu bāzu vadības sistēmu atbalsta izpēti" progresa pārskats

Pārskats Nr. 14 par periodu no 2014.gada 1.janvāra līdz 2014.gada 30.jūnijam.

SATURS

1 Kopsavilkums

Pārskata periodā (2014-01-01 – 2014-06-30.) projekta „Multi - modeļu izstrādes tehnoloģija .NET pielietojumu projektiem" aktivitātes Nr.3.3 "Datu bāzes meta modeļa izstrāde un pētniecība, kas ietver dažādu datu bāzu vadības sistēmu atbalsta izpēti" ietvaros veikti šādi darbi:

1. Datubāžu vadības sistēmu analīze (datu tipu atšķirības, DDL atšķirības, join teikumu sintakse, SQL funkcijas):
   - Oracle;
   - MySQL;
   - Microsoft SQL Server;
   - PostgreSQL;
   - DB2;
   - MS Access;
   - SQLite.

2. Datubāzes metamodeļa prasību analīze.

3. Datubāžu vadības sistēmu atbalsts .NET.

4. Datubāzes meta modeļa izstrāde, tai skaitā, Table, Column, Binding, BindingPair metaentīšu izstrāde.

2 Ievads

Šis pārskats ir veltīts projekta apakšaktivitātes Nr.3.3 "Datu bāzes meta modeļa izstrāde un pētniecība, kas ietver dažādu datu bāzu vadības sistēmu atbalsta izpēti" ietvaros veiktajiem pētījumiem.

Sākotnēji tika veikta datu bāzu vadības sistēmu analīze – datu tipu atšķirības, automātiskās atslēgas ģenerēšanas iespējas, SQL funkcijas un operācijas. Papildus tika veikta dažādu datubāžu vadību sistēmu atbalsts .NET izpēte.

Aktivitātes ietvaros identificētas prasības datubāžu metamodelim, un pētījumu rezultātā izstrādāts tehnoloģiskais metamodelis datubāžu tabulām.

3 Datubāžu vadību sistēmas un to atšķirības

Datubāžu vadību sistēma ir svarīga informatīvo sistēmu (IS) daļa. Mūsdienās IS parasti izmanto relāciju datubāžu vadības sistēmas. Dažādi izstrādātāji piedāvā dažādas datubāžu vadības sistēmas (DBVS). Nākošajā tabulā ir apkopotas populārākās relāciju datubāžu vadības sistēmas [1]:

DBVS	**Izstrādātājs**	**OS (Win, Linux)**	**Licence**
Oracle	Oracle Corporation	Win, Linux	Commercial
MySQL	Oracle Corporation	Win, Linux	GPL v2 vai Commercial
Microsoft SQL Server	Microsoft	Win	Commercial
PostgreSQL	PostgreSQL Global Development Group (open source)	Win, Linux	PostgreSQL License
DB2	IBM	Win, Linux	Commercial
MS Access	Microsoft	Win	Commercial
SQLite	Open source	Win, Linux	SQLite License

Dažādas datubāzes atšķiras gan ar dažādiem tehniskajiem ierobežojumiem, piemēram, maksimālais tabulas izmērs, maksimālais kolonu skaits tabulā, gan arī ar valodu vai valodas dialektu, kuru izmanto, lai vērstos pie datubāzes.
Structured Query Language (SQL) ir valoda, kas dažādu darbību veikšanai ar relāciju datubāzēm. SQL ir standartizēta valoda [2]. SQL standarts attīstās un joprojām tiek papildināts. Ir iznākušas vairākas SQL standarta versijas: SQL-86, SQL-89, SQL-92, SQL:1999, SQL:2003, SQL:2006, SQL:2008. Lai gan praktiski visas datubāžu vadību sistēmu izstrādātāji apgalvo, ka atbalsta SQL standartu, tomēr reālajā dzīvē starp dažādās datubāžu vadības sistēmās izmantoto SQL pastāv atšķirības. Lai saglabātu savietojamību ar iepriekšējām versijām, esošā sintakse netiek mainīta, tāpēc praktiski visās DBVS ir konstrukcijas, kas neatbilst SQL standartam. Tāpat katrai datubāžu vadības sistēmai parasti ir valodas paplašinājumi, kuri neatbilst SQL standartam un kurus var izmantot tikai konkrētajā datubāžu vadības sistēmā. Neapskatīsim visas SQL atšķirības dažādās DBVS. Ar dažiem piemēriem parādīsim, ka dažādās DBVS atšķiras gan datubāžu shēmu definēšana (Data Definition Language vai DDL), gan sintakse datu pieprasījumu teikumos [3].

3.1 Datu tipu atšķirības

Interesanti, bet tāds plaši izmantots datu tips kā BOOLEAN tika ieviests tikai SQL standarta SQL:1999 versijā. Šis datu tips nav atbalstīts Microsoft SQL Server un Oracle, DB2 datubāzēs. Tā vietā Microsoft SQL Server lieto BIT, DB2 - CHAR(1), Oracle - NUMBER(1).

3.2 Automātiskās atslēgas ģenerēšanas atribūts

Praktiski visās populārākajās datubāžu vadību sistēmās atšķiras automātiskās atslēgas ģenerēšanas (autoincrement) atribūts pie primārās atslēgas. SQL standarts nosaka IDENTITY kā automātiskās atslēgas ģenerēšanas atribūtu, bet MySQL izmanto AUTO_INCREMENT, PostgreSQL – SERIAL, MS SQL ir IDENTITY atribūts, bet tā izmantošanai ir savādāka sintakse, ORACLE kā automātiskās atslēgas ģenerēšanas specificēšanai izmanto pavisam savādāku sintaksi.
Automātiskas atslēgas ģenerēšanas iespēja bieži tiek izmantota programmējot IS, un šīs atšķirības rada arī atšķirības DDL.

3.3 SQL funkcijas un operācijas

Atšķirīgas ir arī dažādās datubāzēs izmantoto SQL funkciju un operāciju kopas un funkciju nosaukumi. Minēšu tikai 2: funkcija simbolu virknes garuma noteikšanai un konkatenācijas operators. SQL standartā simbolu virknes garumu nosaka ar funkcijas CHARACTER_LENGTH palīdzību. MS SQL Server tā vietā izmanto LEN un DATALENGTH, arī ORACLE izmanto LEN funkciju.
SQL standartā konkatenācijai izmanto operatoru ||. MySQL šim operatoram ir pavisam cita nozīme - OR (vai operators). Microsoft SQL Server konkatenācijai izmanto +.
Šīs atšķirības rada arī atšķirības datu pieprasījumu teikumos dažādās DBVS.

4 Dažādu datubāžu vadību sistēmu atbalsts .NET

Lai piekļūtu datubāzēm .NET izmanto ADO.NET Data Provider modeli. Dažādi datu sniedzēji (data providers) var tikt izmantoti .NET vidē lai piekļūtu datubāzēm [4]. ADO.NET Data Provider modelis piedāvā stingri definētu interfeisu, lai varētu pieslēgties un darboties ar datubāzi. .NET Framework satur sevī ADO.NET datu sniedzējus, tiešai piekļuvei Microsoft SQL serverim, ka arī piekļuvei citam datu bāzēm, izmantojot ODBC un OLE DB draiverus.
Praktiski visām populārākajām DBVS ir ADO.NET ir savi datu sniedzēji[5]:

DBVS	**Datu sniedzējs**	**Piezīmes**
Oracle	Oracle Data Provider for .NET	
MySQL	MySQL Connector/NET	
Microsoft SQL Server	.NET Framework Data Provider for SQL Server	Ietilpst .NET Framework sastāvā
PostgreSQL	Npgsql	
DB2	ADO.NET data provider for DB2 (IBM)	
SQLite	ADO.NET data provider for SQLite (Phoenix Software Solutions)	

4.1 ADO.NET Entity Framework

ADO.NET Entity Framework ir tehnoloģiju kopa ADO.NET, datu orientētu lietojumprogrammu radīšanai. Tā ir virsbūve virs ADO.NET Data Provider modeļa, kas ļauj lietot Entity Framework ar jebkuru datubāzes vadības sistēmu, kurai ir atbilstošs ADO.NET datu sniedzējs. Galvenā ideja ir tā, lai IS izstrādātājs varētu domāt augstākā abstrakcijas līmeni, t. i. strādāt ar objektiem un to īpašībām, neinteresējoties par datubāzes realizāciju.
ADO.NET Entity Framework iespējas vairāk tiks izpētītas projekta gaitā.

5 Prasības datubāžu metamodelim

Lai atdalītu biznesa loģiku no izstrādes specifikas informatīvo sistēmu izstrādē, mūsu pieeja ir veidot loģiskos un tehnoloģiskos modeļus [6]. Lai varētu veidot modeļus, nepieciešams definēt metamodeļus. Loģiskajiem modeļiem jābūt savam metamodelim, tehnoloģiskajiem - savam.

Ar loģisko metamodeli būtu jāstrādā biznesa analītiķiem, ar tehnoloģisko programmētājiem, kuri programmē informatīvās sistēmas datubāzes pieeju. Šajā dokumentā sīkāk apskatīsim datubāzes tehnoloģisko metamodeli.

Kā tika konstatēts iepriekšējās nodaļās, datubāžu vadības sistēmas atšķiras. Atšķiras gan datubāžu shēmu definēšanas sintakse, gan SQL datu pieprasījumu sintakse. Tehnoloģiskajam metamodelim būtu jābūt tādam, lai ar tā palīdzību varētu veidot modeļus,kas apraksta datubāzi neatkarīgi no izvēlētās datubāžu vadības sistēmas un neatkarīgi no izvēlētas realizācijas tehnoloģijas.

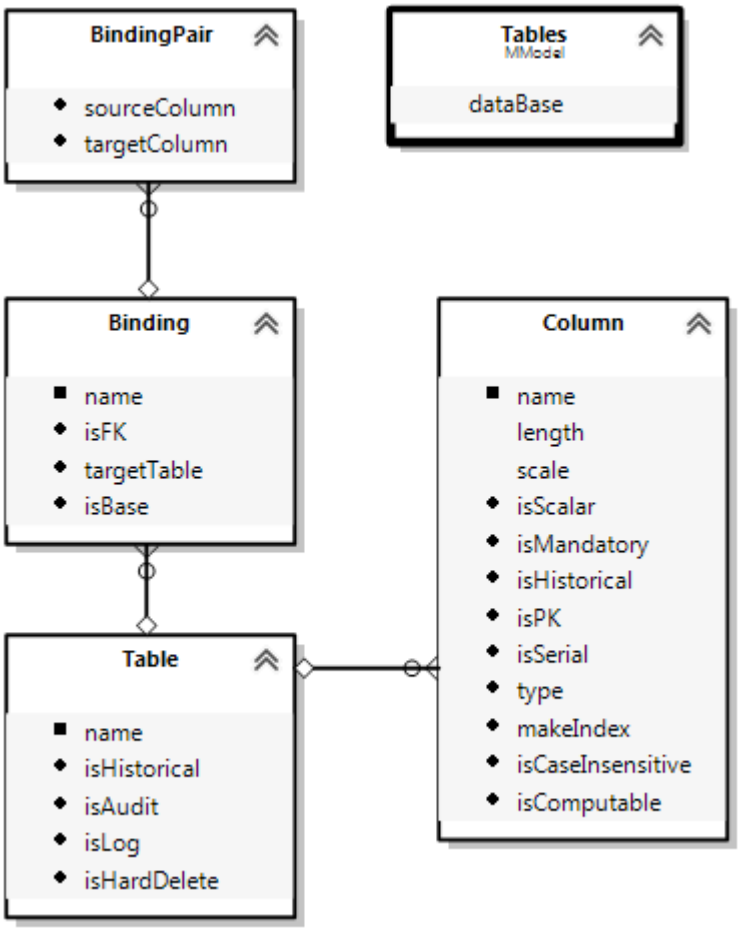
Izmantotās DBVS un tehnoloģijas specifika varētu parādīties tikai koda ģenerācijā no tehnoloģiskā modeļa. Gadījumā, ja tiktu izmantota ADO.NET Entity Framework vai līdzīga tehnoloģija, kas "noslēpj" DBVS specifiku, varētu arī tikt ģenerēts kods ar atbalstu vairākām DBVS.

No modeļa vajadzētu būt iespējai ģenerēt datubāzes shēmas definīciju (DDL teikumus), CRUD (Create, Read, Update, Delete) pieejas metodes datu bāzu tabulu līmenī. Datubāzes metamodelim vajadzētu nodrošināt iespējas:

- definēt saliktas primārās atslēgas;
- specificēt, kurām tabulām veikt auditu;
- specificēt, kurām tabulām veikt visu izmaiņu vēstures saglabāšanu;
- specificēt, kurām tabulām veikt fizisko dzēšanu, kurām ierakstus tikai atzīmēt kā dzēstus.

6 Datubāžu tabulu metamodelis tehnoloģiskajam modelim

Zīmējumā redzams datubāžu tabulu metamodelis tehnoloģiskajam modelim ar vārdu Tables. Izmantojot datubāzes tabulu metamodeli, modelī var aprakstīt datubāzes tabulas, kolonas un relācijas starp tabulām.



Zīmējums 1.

Metamodelim ir metaatribūts:

Vārds	Datu tips	Apraksts
dataBase	string	Datubāzes vārds.

Tālāk tiks aprakstīta katra metaentītijas instance un tai piederošo metaatribūtu instances.

6.1 Table

Table reprezentē datu bāzes tabulu

Vārds	Datu tips	Apraksts
name	string	Tabulas vārds datubāzē.
isHistorical	bool	Ja „true”, atbilstošajai tabulai datubāzē tiek saglabāta visa izmaiņu vēsture.
isAudit	bool	Ja „true”, atbilstošajai tabulai datubāzē tiek veikta auditēšana.
isLog	bool	Ja „true”, par darbībām ar atbilstošo tabulu datubāzē tiek veikta žurnalēšana.
isHardDelete	bool	Ja „true”, tabulai tiek veikta fiziska ierakstu dzēšana, pretējā gadījumā dzēšot ierakstu tabulā, tas netiek fiziski dzēsts, bet gan atzīmēts kā dzēsts.

6.2 Column

Column reprezentē tabulas kolonu.

Vārds	Datu tips	Apraksts
name	string	Kolonas vārds.
length	string	Lauka garums kolonā. Vajadzīgs tikai datu tipiem, kuriem garums tiek uzrādīts.
scale	string	Ciparu skaits aiz komata (tikai kolonām, kurām lauki ir ar decimālo datu tipu).
isScalar	bool	Ja „true”, lauks kolonā nepieder pie ārējās atslēgas.
isMandatory	bool	Ja „true”, lauks kolonā ir obligāts.
isHistorical	bool	Ja „true”, laukam ir jāsaglabā tā izmaiņu vēsture.
isPK	bool	Ja „true”, lauks ietilpst primārajā atslēgā.
isSerial	bool	Ja „true”, lauks ir autoincrement lauks.
type	string	Lauka datu tips.
makeIndex	bool	Ja „true”, kolona ir indeksēta.
isCaseInsensitive	bool	Ja „true”, lauks nav reģistrjūtīgs, pretējā gadījumā reģistrjūtīgs.
isComputable	bool	Ja „true”, lauks ir izrēķināms (nav datubāzē).

6.3 Binding

Binding reprezentē relācijas starp tabulām.

Vārds	Datu tips	Apraksts
name	string	Relācijas vārds.
isFK	bool	Ja „true”, relācijai datubāzē tiek definēta ārējā atslēga.
targetTable	string	Ar relāciju saistītās tabulas vārds.
isBase	bool	Ja „true”, relācija Ir “mantošanas” relācija.

6.4 BindingPair

BindingPair nodrošina iespēju veidot relācijas, kas saistītas ar vairākiem laukiem, t. i. nodrošina saliktas atslēgas.

Vārds	Datu tips	Apraksts
sourceColumn	string	Relācijas vārds.
targetColumn	bool	Ja „true”, datubāzē relācijai tiek definēta ārējā atslēga.

7 Rezultāti

Aktivitātes ietvaros veiktā darba rezultātā tika izpētītas dažādas datu bāzu vadības sistēmas un izstrādāts datubāžu tabulu metamodelis tehnoloģiskajam modelim ar vārdu Tables. Izmantojot datubāzes tabulu metamodeli, modelī var aprakstīt datubāzes tabulas, kolonas un relācijas starp tabulām.

8 Literatūras saraksts

[1] DB-Engines Ranking http://db-engines.com/en/ranking
[2] SQL http://en.wikipedia.org/wiki/SQL
[3] Comparison of different SQL implementations http://troels.arvin.dk/db/rdbms/
[4] ADO.NET Data Providers http://msdn.microsoft.com/en-us/data/dd363565.aspx
[5] .NET Framework Data Providers http://msdn.microsoft.com/en-us/library/a6cd7c08%28v=vs.110%29.aspx
[6] Projekta aktivitātes Nr.3.2 "Biznesa lēmumu atdalīšana no izstrādes lēmumiem" progresa pārskats.