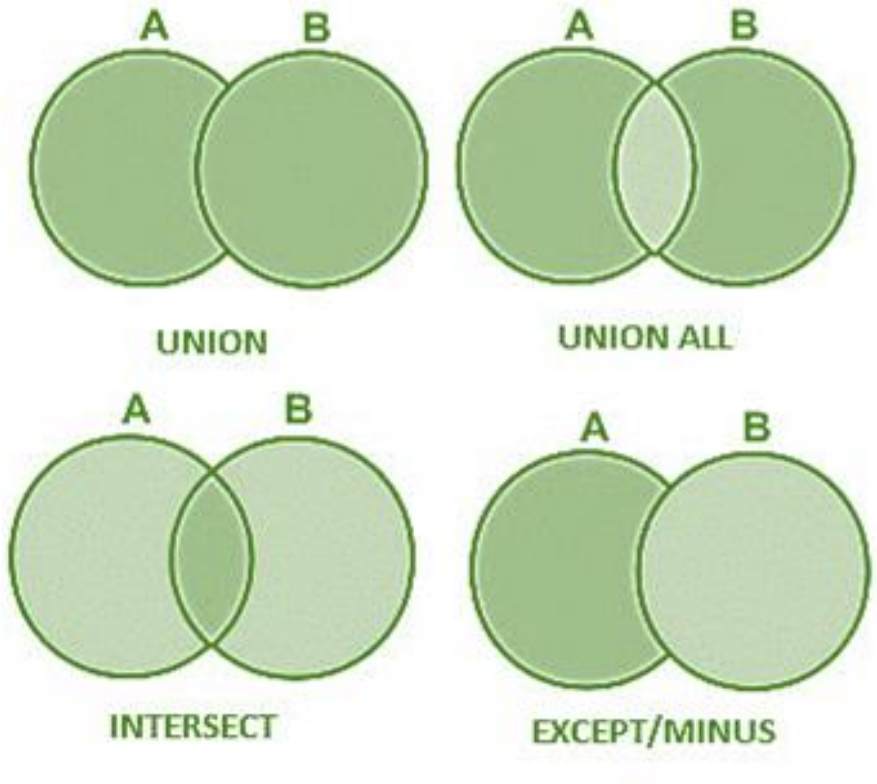
ktu informatikos fakultetas 1922

DML, virtualios lentelės

VIEW


A
B
UNION
A
B
UNION ALL
A
B
INTERSECT
A
B
EXCEPT/MINUS

Sumos realizacija *SQL*

KTU

PAVARDE	VARDAS	APGYNE
Petraitis	Petras	1990
Jonaitis	Jonas	1995
Antanaitis	Antanas	1990

VU

PAV	VAR	APG
Petraitis	Petras	1990
Zuokys	Jonas	1995
Antanaitis	Antanas	1993

KTU_VU

PAVARDE	VARDAS	APGYNE
Petraitis	Petras	1990
Jonaitis	Jonas	1995
Antanaitis	Antanas	1990
Petraitis	Petras	1990
Zuokys	Jonas	1995
Antanaitis	Antanas	1993

```
(SELECT * FROM ktu)
UNION [DISTINCT] [ALL]
(SELECT pav AS pavarde, var AS
vardas, apg AS apgyne FROM vu) ;
```

```
(SELECT *, ‚ktu‘ AS vieta FROM
ktu)
UNION [DISTINCT] [ALL]
(SELECT pav AS pavarde, var AS
vardas, apg AS apgyne, ‚vu‘ AS
vieta FROM vu) ;
```

Sankirtos realizacija *SQL*

KTU

PAVARDE	VARDAS	APGYNE
Petraitis	Petras	1990
Jonaitis	Jonas	1995
Antanaitis	Antanas	1990

VU

PAVARDE	VARDAS	APGYNE
Petraitis	Petras	1990
Zuokys	Jonas	1995
Antanaitis	Antanas	1993

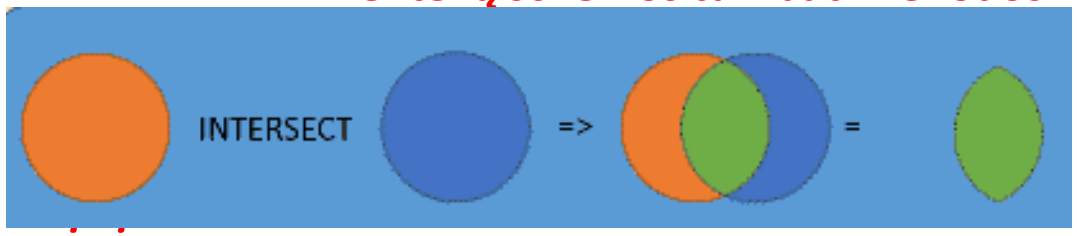
Užklausa: Kurie asmenys studijavo ir KTU, ir VU . Pateikite tik asmenų pavardes ir vardus.

KTU_IR_VU

PAVARDE	VARDAS
Petraitis	Petras
Antanaitis	Antanas

```
(SELECT pavarde, vardas FROM ktu)
INTERSECT
(SELECT pavarde, vardas FROM vu) ;
```

Lentelių schemos turi būti vienodos




```
SELECT DISTINCT id
FROM t1 INNER JOIN t2
 USING(id);
```

MySQL realizuoja per inner join

Skirtumo realizacija *SQL*

KTU

PAVARDE	VARDAS	APGYNE
Petraitis	Petras	1990
Jonaitis	Jonas	1995
Antanaitis	Antanas	1990

VU

PAVARDE	VARDAS	APGYNE
Petraitis	Petras	1990
Zuokys	Jonas	1995
Antanaitis	Antanas	1993

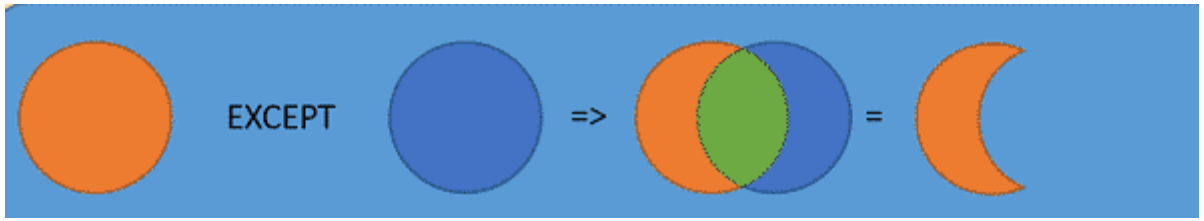
Užklausa: Kurie asmenys studijavo tik KTU . Pateikite tik asmenų pavardes ir vardus.

TIK_KTU

PAVARDE	VARDAS
Jonaitis	Jonas

```
(SELECT pavarde, vardas FROM ktu)
EXCEPT
(SELECT pavarde, vardas FROM vu) ;
```

Lentelių schemos turi būti vienodos



MySQL realizuojama per left join su papildoma sąlyga

ktu informatikos fakultetas 1922

Vidinių užklausų realizacija *SQL*

Vidinės užklausos su pavienėmis grąžinamomis reikšmėmis

FILMAI (pavadinimas, metai, trukmė, spalvotas, studija, prodiuseris)
PRODIUSERIAI (kodas, vardas, adresas, pelnas)

Užklausa: *Kas yra filmo "Star Wars" prodiuseris?*

```
SELECT vardas
FROM filmai INNER JOIN prodiuseriai ON filmai.prodiuseris = prodiuseriai.kodas
WHERE pavadinimas = 'Star Wars';
```

Vidinė užklausa →

```
1) SELECT vardas
2) FROM prodiuseriai
3) WHERE kodas =
4)     (SELECT prodiuseris
5)      FROM filmai
6)          WHERE pavadinimas = 'Star Wars'
       );
```

Vidinės užklausos su grupe grąžinamų reikšmių

Jei vidinė užklausa grąžina daugiau nei vieną reikšmę, tai norint jas palyginti reikia naudoti operatorius IN, ANY, ALL, EXISTS.

Tarkime, kad turime kortežą ***s*** ir santykį ***R***.

1. ***s* IN *R*** yra teisinga tik tada, jei s yra lygus bent vienam kortežui santykyje R.
2. ***s* NOT IN *R*** yra teisinga tik tada, jei s nėra lygus nei vienam kortežui santykyje ***R***.
3. ***s* > ALL *R*** yra teisinga tik tada, jei s yra didesnis už visus kortežus santykyje ***R***. *Vietoje '>' gali būti panaudotas bet kuris kitas palyginimo operatorius.* ***s* <> *ALL R*** *ekvivalentiškas* ***s* NOT *IN R***.
4. ***s* > ANY *R*** yra teisinga tik tada, jei s yra didesnis bent už vieną kortežą santykyje ***R***. *Vietoje '>' gali būti panaudotas bet kuris kitas palyginimo operatorius.* ***s* = *ANY R*** *ekvivalentiškas* ***s* IN *R***.
5. **EXISTS *R*** yra teisinga tik tada, jei ***R*** nėra tuščias.

Vidinė užklausa. Predikatas **IN**

FILMAI (pavadinimas, metai, trukmė, spalvotas, studija, prodiuseris)
ATLIKEJAI (aktoriaus_vard, vaidmuo, filmo_pav, filmo_metai)
PRODIUSERIAI (kodas, vardas, adresas, pelnas)

Užklausa: *Raskite Harrison'o Ford'o prodiuserius?*

```
1) SELECT vardas
2) FROM prodiuseriai
3) WHERE kodas IN
4)     (SELECT prodiuseris
5)      FROM filmai
6)          WHERE (pavadinimas, metai) IN
7)              (SELECT filmo_pav, filmo_metai
8)               FROM atlikejai
9)               WHERE aktoriaus_vard = 'Harrison Ford'
                 )
         );
```

s* IN *R yra teisinga tik tada, jei s yra lygus bent vienam kortežui santykyje R

Ta pati užklausa be vidinių užklausų

FILMAI (pavadinimas, metai, trukmė, spalvotas, studija, prodiuseris)
ATLIKEJAI (aktoriaus_vard, vaidmuo, filmo_pav, filmo_metai)
PRODIUSERIAI (kodas, vardas, adresas, pelnas)

Užklausa: *Raskite Harrison'o Ford'o prodiuserius?*

```
SELECT vardas
FROM prodiuseriai INNER JOIN  filmai  ON kodas = prodiuseris INNER JOIN atlikejai ON
(pavadinimas = filmo_pav  AND metai = filmo_metai)
WHERE  aktoriaus_vard = 'Harrison Ford';
```

Vidinė užklausa. Predikatas **ANY**

FILMAI (pavadinimas, metai, trukmė, spalvotas, studija, prodiuseris)
PRODIUSERIAI (vardas, adresas, kodas, pelnas)

Užklausa: *Raskite prodiuserius, kurie uždirbo mažiau bent už vieną prodiuserį , kuris sukūrė bent vieną filmą 1990 metais?*

```
1) SELECT vardas
2) FROM prodiuseriai
3) WHERE pelnas < ANY
4)     (SELECT pelnas
5)      FROM prodiuseriai INNER JOIN filmai ON kodas = prodiuseris
6)      WHERE metai = 1990 );
```

s* > ANY *R yra teisinga tik tada, jei s yra didesnis bent už vieną kortežą santykyje ***R***.

Vidinė užklausa. Predikatas **ALL**

FILMAI (<u>pavadinimas, metai</u>, trukmė, spalvotas, studija, prodiuseris)
PRODIUSERIAI (vardas, adresas, kodas, pelnas)

Užklausa: *Raskite prodiuserius, kurie uždirbo mažiau nei tie prodiuseriai, kurie sukūrė bent vieną filmą 1990 metais?*

```
1) SELECT vardas
2) FROM prodiuseriai
3) WHERE pelnas < ALL
4)     (SELECT pelnas
5)      FROM prodiuseriai INNER JOIN  filmai ON  kodas = prodiuseris
6)      WHERE metai = 1990 );
```

s* > ALL *R yra teisinga tik tada, jei s yra didesnis už visus kortežus santykyje ***R***.

Vidinė užklausa. Predikatas **EXISTS**

ATLIKEJAI (filmo_pav, filmo_metai, vaidmuo, aktoriaus_vard)
AKTORIAI (vardas, adresas, lytis, gimimo_data)

Užklausa: *Raskite aktorius (vardas, adresas, gimimo data), kurie vaidino bent viename filme, sukurtame 1990 metais?*

```
1) SELECT vardas, adresas, gimimo_data
2) FROM aktoriai
3) WHERE EXIST
4)     (SELECT vardas, adresas, gimimo_data
5)      FROM atlikėjai INNER JOIN aktoriai ON vardas = aktoriaus_vard
6)      WHERE metai = 1990 );
```

EXISISTS *R* yra teisinga tik tada, jei ***R*** nėra tuščias

Koreliuotosios vidinės užklausos

FILMAI (pavadinimas, metai, trukmė, spalvotas, studija, prodiuseris)

Užklausa: *Raskite pavadinimus, kurie buvo panaudoti daugiau, nei viename filme?*

```
1) SELECT pavadinimas
2) FROM filmai as kiti_filmai
3) WHERE metai <> ANY
4)     (SELECT metai
5)      FROM filmai
6)   WHERE filmai.pavadinimas = kiti_filmai.pavadinimas
7)   and filmai.metai <> kiti_filmai.metai ;
```

s* < ANY *R yra teisinga tik tada, jei s yra mažesnis bent už vieną kortežą santykyje ***R***.

Vidinės užklausos FROM dalyje

FILMAI (pavadinimas, metai, trukmė, spalvotas, studija, prodiuseris)
ATLIKEJAI (filmo_pav, filmo_metai, vaidmuo, aktoriaus_vard)
PRODIUSERIAI (vardas, adresas, kodas, pelnas)

Užklausa: *Raskite Harrison'o Ford'o prodiuserius?*

```
SELECT vardas
FROM prodiuseriai INNER JOIN (SELECT prodiuseris
                FROM filmai INNER JOIN atlikejai ON (pavadinimas =
    filmo_pav and  metai = filmo_metai )
                WHERE aktoriaus_vard = 'Harrison Ford'
                ) AS prod ON prodiuseriai.kodas = prod.prodiuseris ;
```

ktu informatikos fakultetas 1922

Virtualios lentelės

Virtualios lentelės

Virtualios lentelės (angl. view) - tai DB loginė lentelė, atvaizduojanti vienos arba kelių DB lentelių duomenis norimu pjūvių. virtualios lentelės užtikrina vartotojų teisių apribojimą, suteikdamos galimybę naudotis tik virtualioje lentelėje apibrėžtais duomenimis.

CREATE

[ALGORITHM = {MERGE | TEMPTABLE | UNDEFINED}]

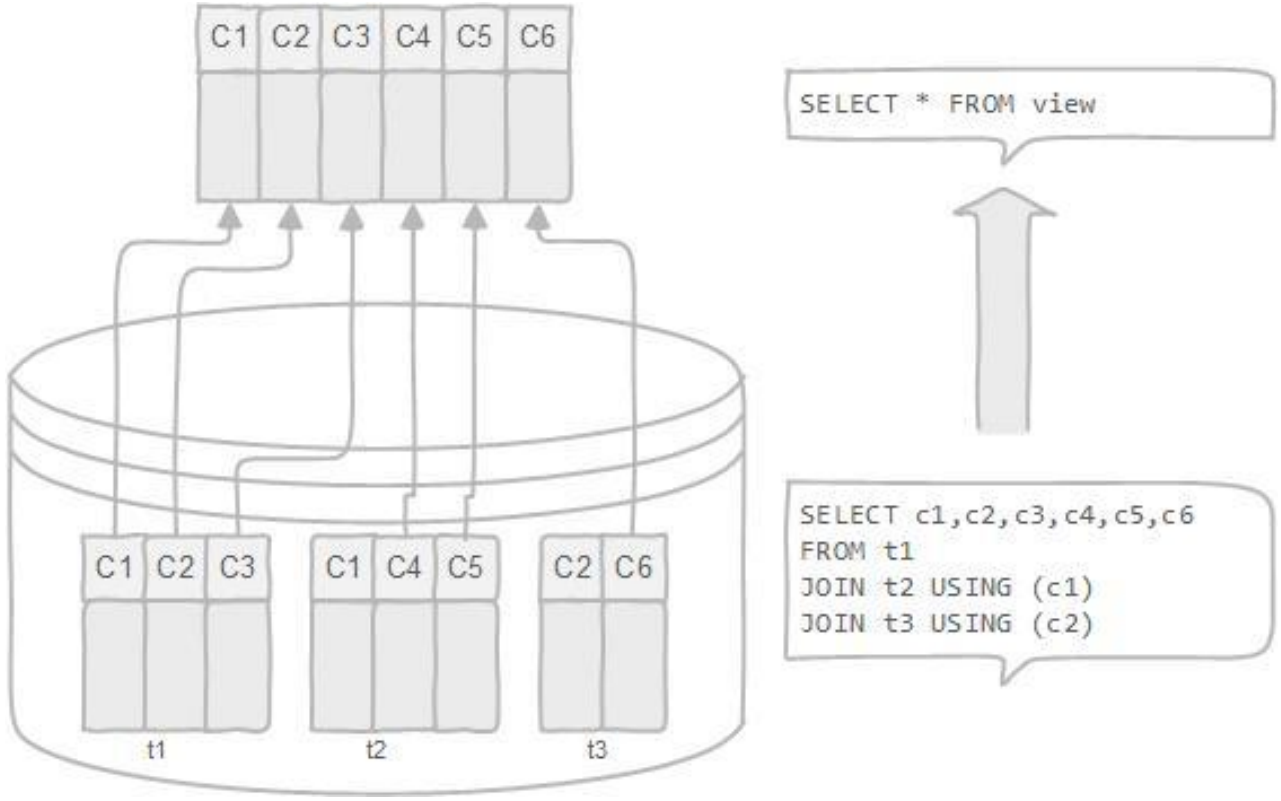
VIEW [database_name].[view_name]

AS

[SELECT statement]

```
CREATE VIEW virtualios lentelės_vardas [(atributas [,…n])]
 [WITH ENCRYPTION]
 AS
 select_sakinys
 [WITH CHECK OPTION]
```

Virtualios lentelės



Kodėl verta naudoti virtualias lenteles

- Padeda supaprastinti sudėtingas užklausas arba paslėpti DB schemos sudėtingumą;
- Suteikia priemones naudotojams apriboti duomenų pasiekimą konkretiems naudotojams;
- Suteikia dar vieną saugumo sluoksnį, ypač kai kalbama apie duomenų redagavimą;
- Suteikia galimybę iškart turėti išvestinius (agreguotus) laukus;
- Virtualios lentelės suteikia galimybę sistemų suderinamumui realizuoti.

Virtulių lentelių minusai

- Našumas lėtokas ypač jei virtualios lentelės konstruojamos kitų virtualių lentelių pagrindu;
- Lentelių priklausomybių palaikymas, jei pasikeičia DB struktūra turėsite pertvarkyti ir virtualias lenteles.

MySQL virtulių lentelių apribojimai

- Virtualioms lentelėms, negalima sukurti indeksų;
- Iki Mysql 5.7.7 versijos nebuvo galim naudoti užklausų užklausose, kai konstruojamos virtualios lentelės;
- MySql nepalaiko materializuotų virtualių lentelių;
- Tik be join virtualios lentelės gali būti redaguojamos.

MySQL virtuli redaguojama lentelė negali turėti

- MIN, MAX, SUM, AVG, COUNT;
- DISTINCT;
- GROUP BY ;
- HAVING;
- UNION arba UNION ALL;
- Left join or outer join;
- Nuorodas į neredaguojamas lenteles.

Virtualių lentelių tipai

Standartinės virtualios lentelės;

Indeksuotos (indexed) virtualios lentelės; (Sukuriamas fizinis indeksas, ir virtuali lentelė saugoma bazėje)

Padalintos virtualios lentelės (partitioned) (Paskirtymas tarp duomenų bazių arba serverių serverių)

- Lokalios – tarp duomenų bazių;
- Paskirstytos – tarp serverių;

atributas

- atributo vardą būtina suteikti tik tuomet, kai jis yra aritmetinės išraiškos rezultatas, funkcija ar konstanta, kai du atributai turi vienodus vardus (Join atveju), kai atributui virtualioje lentelėje norime suteikti kitą vardą, nei jis buvo DB lentelėje.

  Atributų vardus galima nurodyti ir *Select* -sakinyje

  Jei atributų vardų nenurodome, tai jie paveldimi iš *Select*_sakinio.

ktu informatikos fakultetas 1922

Pastaba:

Naudojant **ALTER VIEW** su CREATE VIEW sukurta virtuali lentelė gali būti pakeista, neįtakojant susietų procedūrų ir trigerių, tačiau pakeičiant ankstesnius atributų vardus kitais, pakeičiant apribojimo sąlygas

WITH ENCRYPTION

Užkoduoja sisteminėje lentelėje esantį tekstą su CREATE VIEW išraiška

AS

virtualios lentelės apibrėžimas

ktu informatikos fakultetas 1922

select_sakinys

SELECT sakinys virtualiai lentelei aprašyti. Ji gali susidėti iš daugiau nei vienos lentelės arba kitų viertualių lentelių. Virtualią lentelę sudarančioms lentelėms ir kitoms virtualioms lentelės turi būti nustatytos atitinkamos vartotojo teisės.

Nors SELECT sakinys gali būti bet kokio sudėtingumo, tačiau yra tam tikri apribojimai.

Negalima naudoti:

- ORDER BY, COMPUTE BY operatorių;
- INTO raktinio žodžio;
- laikinos lentelės nuorodos.

SELECT sakinyje galima naudoti:

- funkcijas;
- keletą SELECT sakinių, atskirtų UNION operatoriumi.

WITH CHECK OPTION

užtikrina, kad visos duomenų modifikacijos, atliekamos su virtualia lentele, tenkintų *Select* sakinio kriterijus. Kai duomenys modifikuojami per virtualią lentelę, tai WITH CHECK OPTION garantuoja, kad eilutės duomenys pasiliks matomi ir atlikus pakeitimą.

Pastabos: virtuali lentelė gali būti naudojama modifikavimui , jei:

- *select_sakinys* neturi agregatinių funkcijų *Select* sąraše, o taip pat neturi TOP, GROUP BY, UNION, arba DISTINCT operatorių. Kol reikšmės nemodifikuojamos, agreguotos funkcijos gali būti naudojamos subužklausose FROM sakinyje.
- *select_sakinys* neturi išvestinių laukų *Select* sąraše. Išvestiniai laukai sudaryti naudojant funkcijas, sudėties, atimties ar pan. operacijas.
- FROM operatorius nurodo bent vieną lentelę, o išrenkami duomenys nėra vien tik ne iš lentelės išvedamos reikšmės ar ne lentelių pagrindu suformuojami rezultatai.

Indeksuotos virtualios lentelės

Kai užklausa gražina daug duomenų naudojant agregavimo funkcijas;

Duomenų bazėje išsaugomas lentelių apjungimo ir agregavimo rezultatas;

CREATE VIEW *virtualios lentelės_vardas* [(atributas [,…*n*])] with schemabinding

Create unique clustered index on <poschemes vardas> (< lauko pavadinimas>)

Padalintos virtualios lentelės

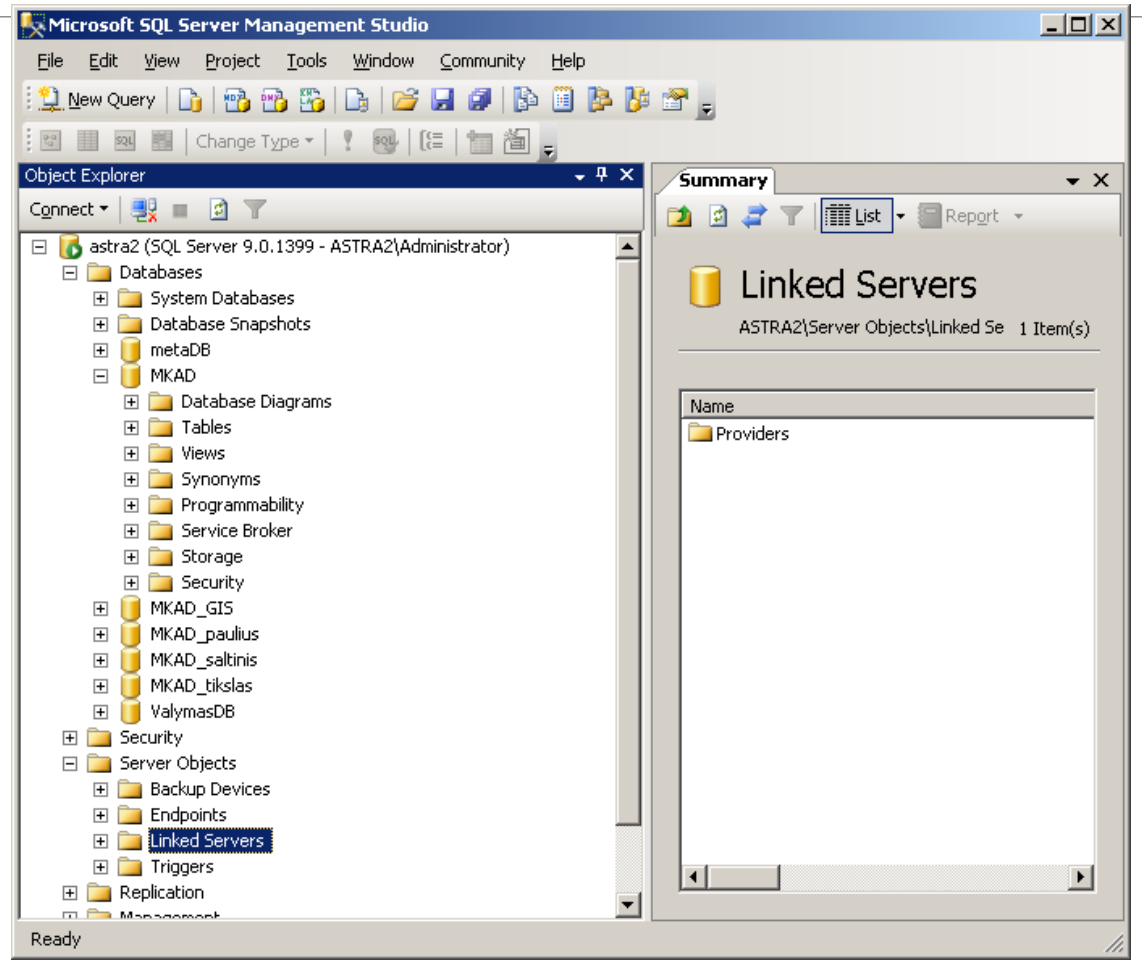
Kai užklausa gražina daug duomenų;

Užklausos vykdymas vyksta ilgai;

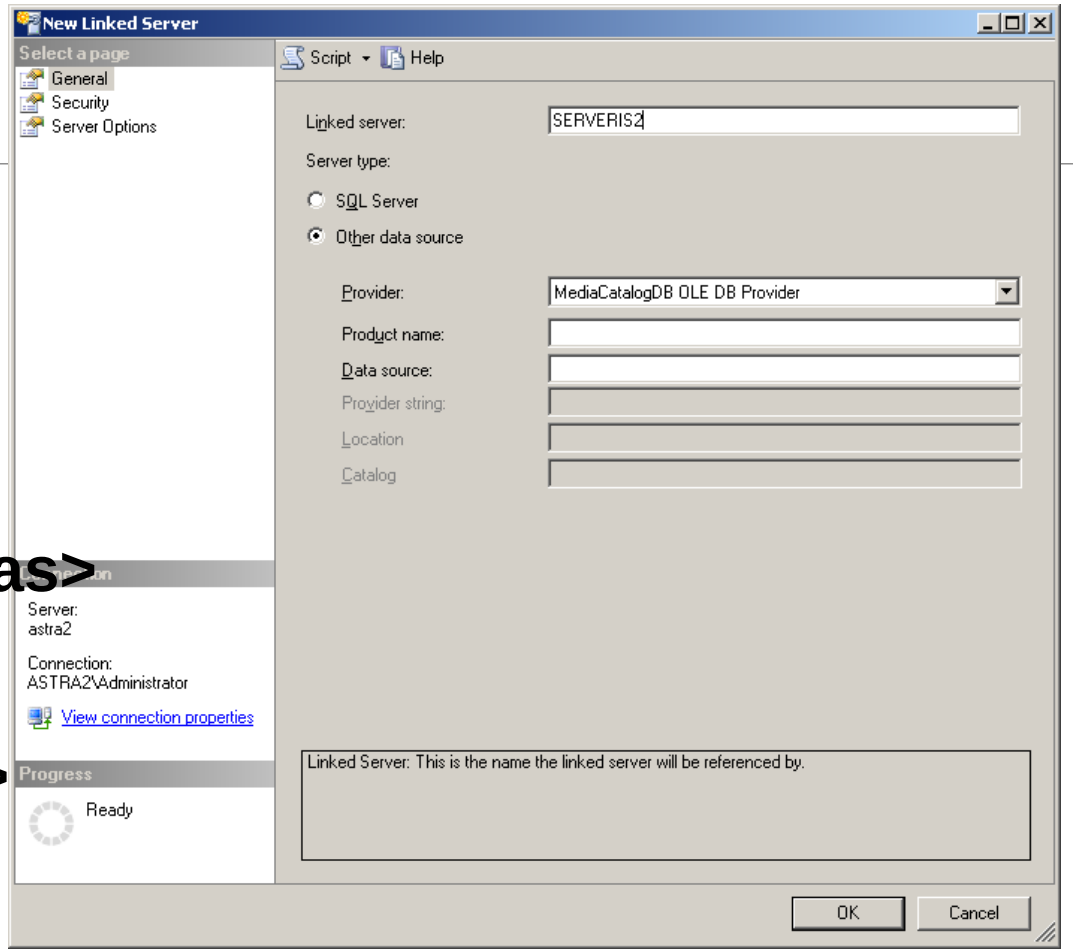
Duomenų apdorojimui naudojamas paskirstytas sprendimas;

Padalintos virtualios lentelės (MS SQL server atveju)

Esant paskirstytai virtualiai lentelei turi būti sukurta sąsaja tarp serverių



Padalintos virtualios lentelės



CREATE VIEW <pavadinimas>

AS

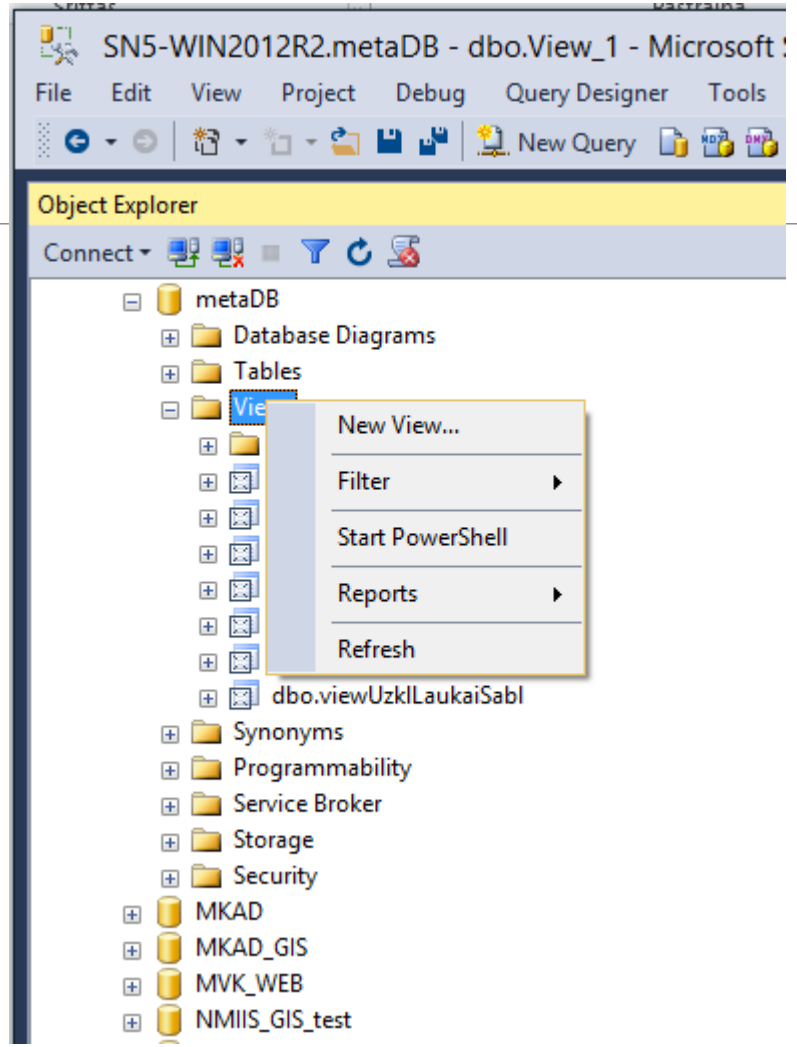
SELECT * FROM <lentele1>

UNION all

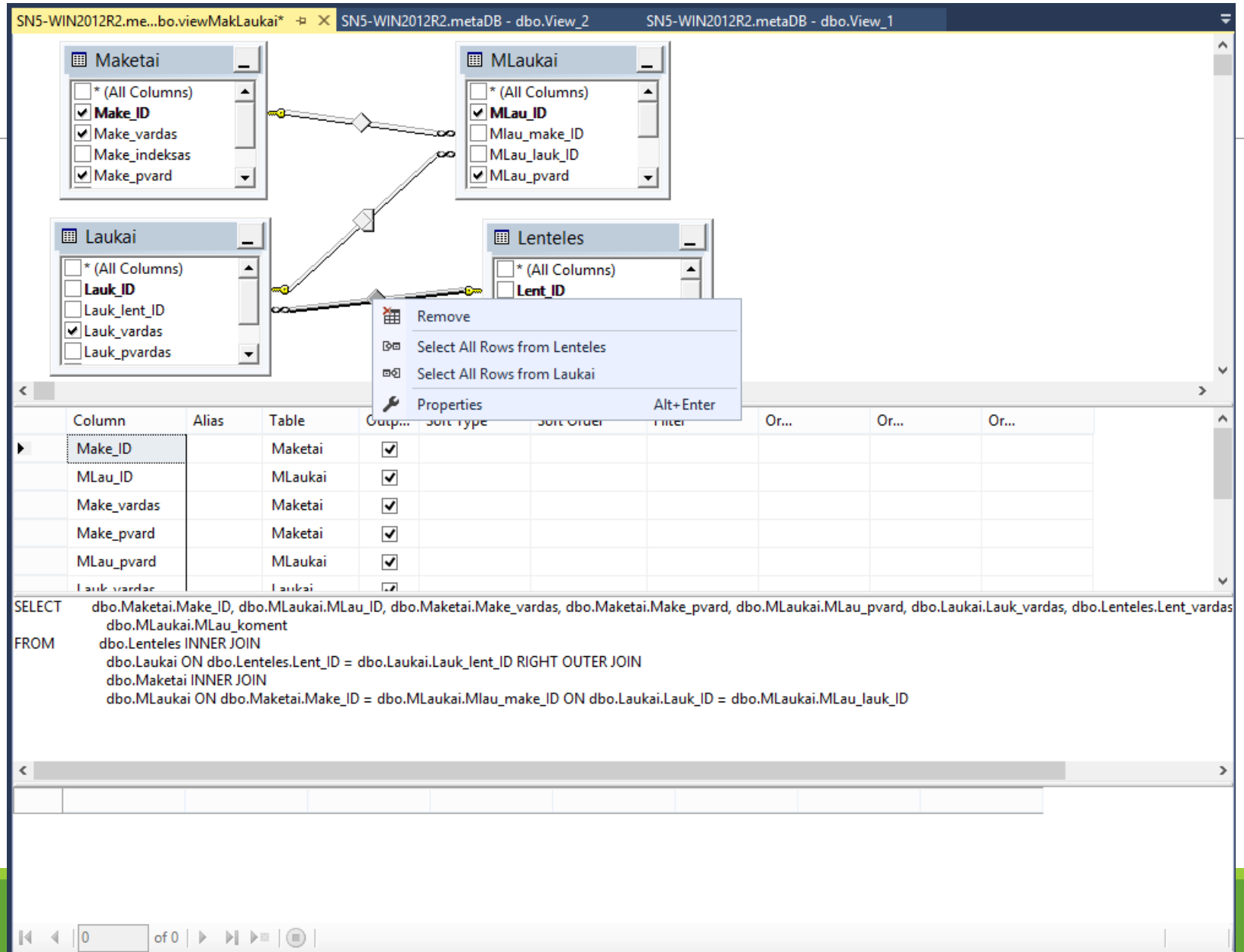
SELECT * from <serveris2>.<duomenu baze2>.<lentelė2>

Virtualios lentelės kūrimas

MS SQL Server Management Studio



Virtualios lentelės kūrimas MS SQL Server Management Studio



Virtualių lentelių sudarymo pavyzdžiai (1)

Kuriama paprasta virtuali lentelė SELECT sakinio pagrindu iš vienos lentelės. Virtuali lentelė naudinga, jei tokia laukų kombinacija dažnai reikalinga.

```
USE pubs
IF EXISTS (SELECT TABLE_NAME FROM
INFORMATION_SCHEMA.VIEWS
WHERE TABLE_NAME = 'titles_view')
DROP VIEW titles_view
GO
CREATE VIEW titles_view
  AS
  SELECT title, type, price, pubdate
  FROM titles
```

Virtualių lentelių sudarymo pavyzdžiai

Kuriama virtuali lentelė su parametru ***WITH ENCRYPTION*** . Panaudojami skaičiuojami laukai ir jų pervardinimas.

```
USE pubs
IF EXISTS (SELECT TABLE_NAME FROM
INFORMATION_SCHEMA.VIEWS
WHERE TABLE_NAME = 'accounts')
DROP VIEW accounts
GO
CREATE VIEW accounts (title, advance, amt_due)
WITH ENCRYPTION
AS
```

```
SELECT title, advance, price * royalty * ytd_sales
FROM titles
WHERE price > $5
GO
```

Toliau pateikiama užklausa užkoduotos virtualios lentelės kūrimo procedūros identifikatoriui ir aprašui (***text***) iškviesti

```
USE pubs
GO
SELECT c.id, c.text
FROM syscomments c, sysobjects o
WHERE c.id = o.id and o.name = 'accounts'
GO
```

ktu informatikos fakultetas 1922

Toliau pateikiamas gautų rezultatų rinkinys:

```
id          text
----------  ----------------------------------------
1925581898  ????????????????????????????????????????
            ????????????????????????????????????????
            ????????????????????????????????????????
            ????????????????????????????????????????

(1 row(s) affected)
```

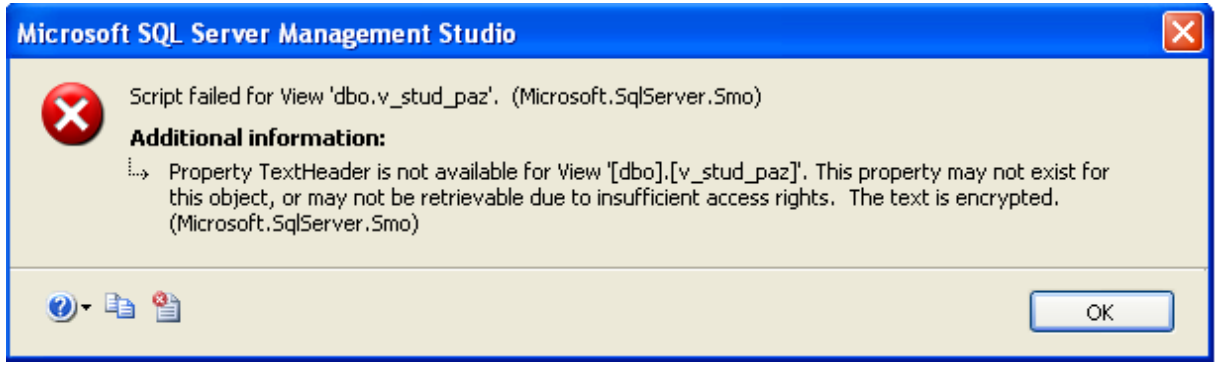
Virtualios lentelės tekstas negali būti perskaitytas, kadangi ji buvo sukurta nurodant užkodavimo parametrą

Šifravimas (encryption)

Naudojama ***with encryption***

CREATE VIEW [dbo].[v_stud_paz] **with encryption** AS SELECT ... FROM ...

Klaidos pranešimas bandant peržiūrėti virtualios lentelės kodą



Sript View As -> Create To -> New Query Editor Window

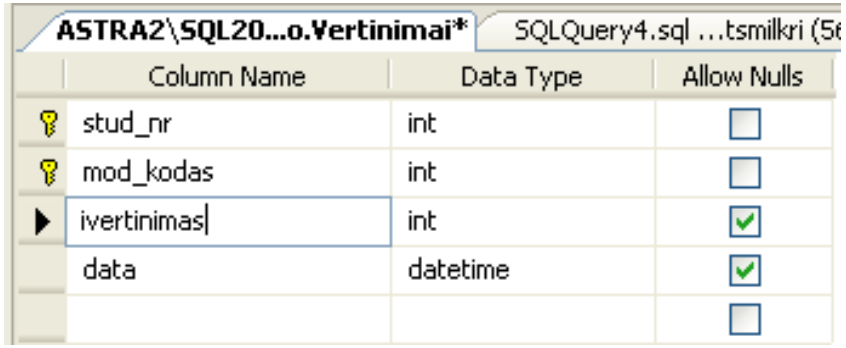
Virtualių lentelių sudarymo pavyzdžiai

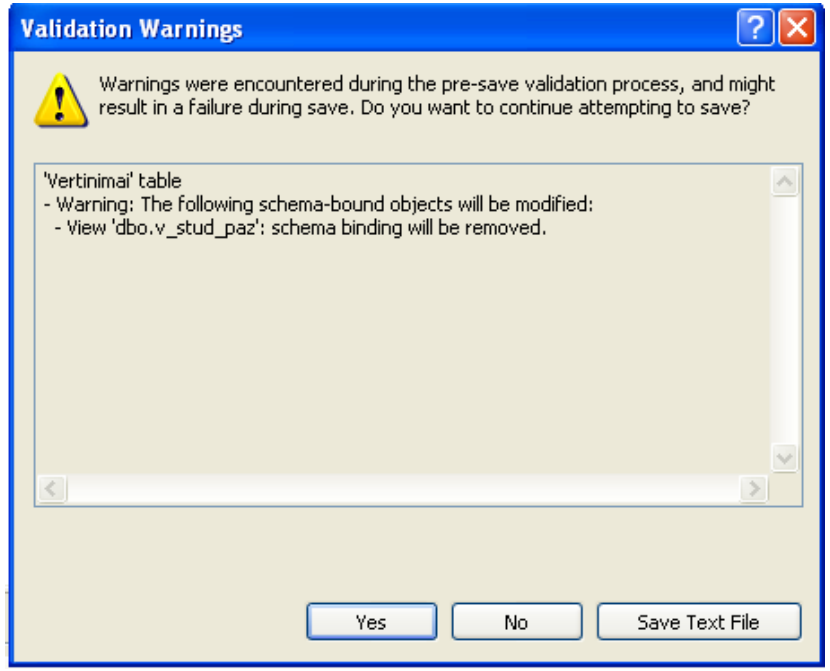
```
CREATE VIEW CAonly
AS
SELECT au_lname, au_fname, city, state
FROM authors
WHERE state = 'CA'
WITH CHECK OPTION
GO
```

Susiejimas su DB schema (angl. schemabinding)

Teikia pranešimus apie bandomą redaguoti lentelių struktūrą. Susiejami atitinkami virtualios lentelės ir fizinės lentelės stulpeliai ir juos redaguojant gaunamas perspėjimas.

CREATE VIEW [dbo].[v_stud_paz] **with schemabinding** AS
SELECT ... FROM ...





Virtualių lentelių sudarymo pavyzdžiai

Panaudojama funkcija @@ROWCOUNT

```
USE pubs
IF EXISTS (SELECT TABLE_NAME FROM
INFORMATION_SCHEMA.VIEWS
WHERE TABLE_NAME = 'myview')
DROP VIEW myview
GO
CREATE VIEW myview
AS
SELECT au_lname, au_fname, @@ROWCOUNT AS bar
FROM authors
WHERE state = 'UT'
```

Virtualių lentelių sudarymo pavyzdžiai

Virtualioje lentelėje apjungiamos 4 lentelės **SUPPLY1**, **SUPPLY2**, **SUPPLY3**, ir **SUPPLY4**, atitinkančios keturių filialų naudojamas tiekėjų lenteles

```
CREATE VIEW VISI_TIEKEJAI
AS
SELECT * FROM SUPPLY1
UNION
SELECT * FROM SUPPLY2
UNION
SELECT * FROM SUPPLY3
UNION
SELECT * FROM SUPPLY4
```