


1

# 1 LEQC

# MS SQL Server-ის საინსტალაციოს გადმოწერა


Try SQL Server on-premises or in the cloud



**SQL Server on Azure**

Run SQL Server on Azure SQL with built-in security and manageability.


Get started



**SQL Server at the edge**

Extend SQL to IoT devices for real-time analysis with Azure SQL Edge.

Get started




**SQL Server on-premises**

Build intelligent, mission-critical applications with a scalable, hybrid data platform.

Free trial


Or, download a free specialized edition



**Developer**

SQL Server 2019 Developer is a full-featured free edition, licensed for use as a development and test database in a non-production environment.

Download now >



**Express**

SQL Server 2019 Express is a free edition of SQL Server, ideal for development and production for desktop, web, and small server applications.

Download now >

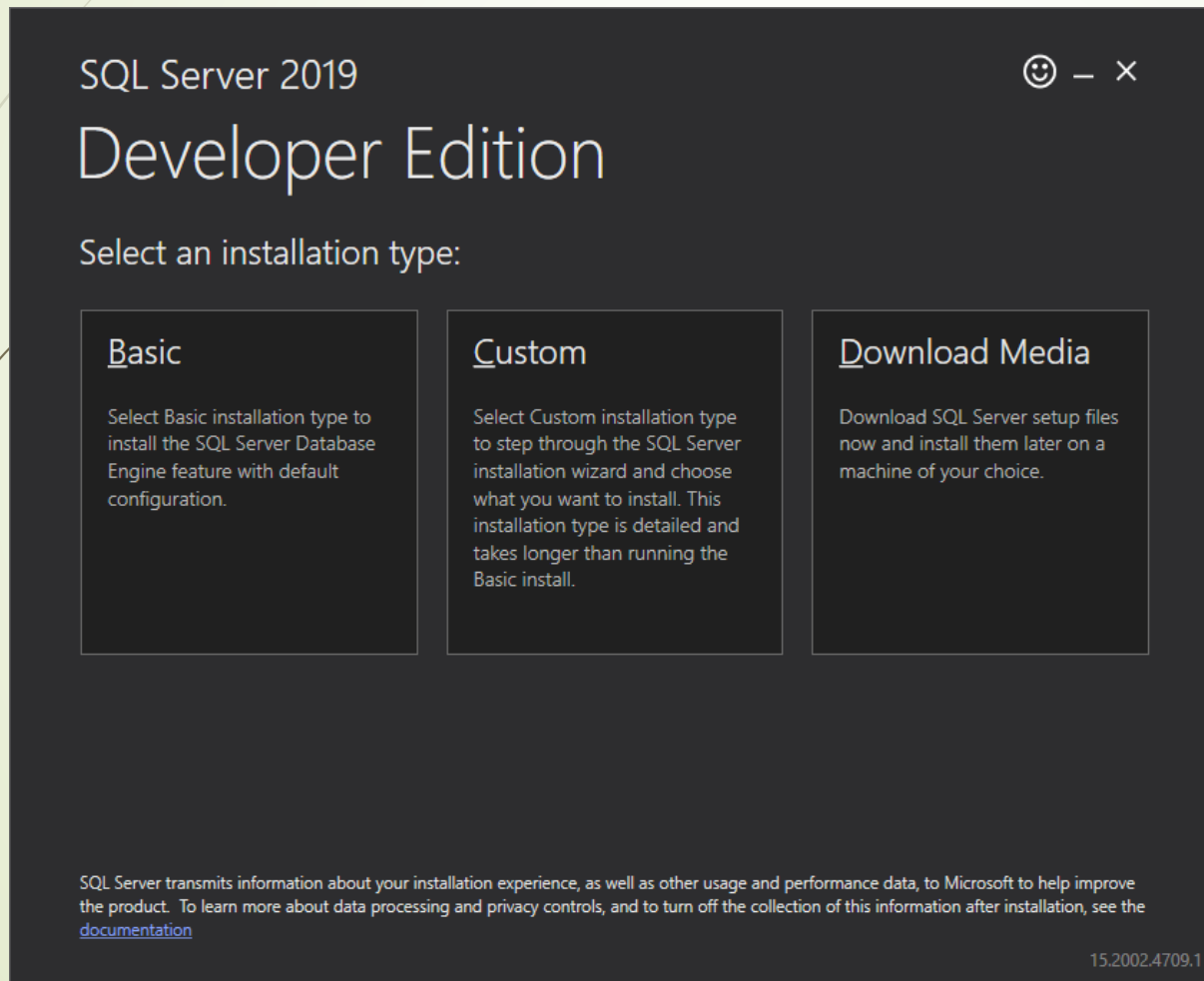
დღევანდელ დღეს საუკეთესო ვარიანტია MS SQL Server 19 Developer Edition, რომელიც არის ყველასათვის ხელმისაწვდომი მონაცემთა ბაზების წარმოების მძლავრი საშუალება და სრულიად უფასო ლიცენზიით - ერთადერთი შეზღუდვა არის ლიცენზიის შესყიდვის გარეშე არ უნდა გამოიყენოთ კომერციული საქმიანობისათვის. ამ ვერსიის ჩამოსაწერად შეგვიძლია გამოვიყენოთ ლინკი:

<https://www.microsoft.com/en-us/sql-server/sql-server-downloads>

და მაუსის მარცხენა ღილაკით გავააქტიუროთ „Download now“

# MS SQL Server-ის საინსტალაციოს გადმოწერა

გააქტიურების შემდგომ ჩამოიტვირთება ფაილი „SQL2019-SSBI-Dev.exe“, რომელიც უნდა გაეშვას სისტემაში ადმინისტრატორის უფლებებით.

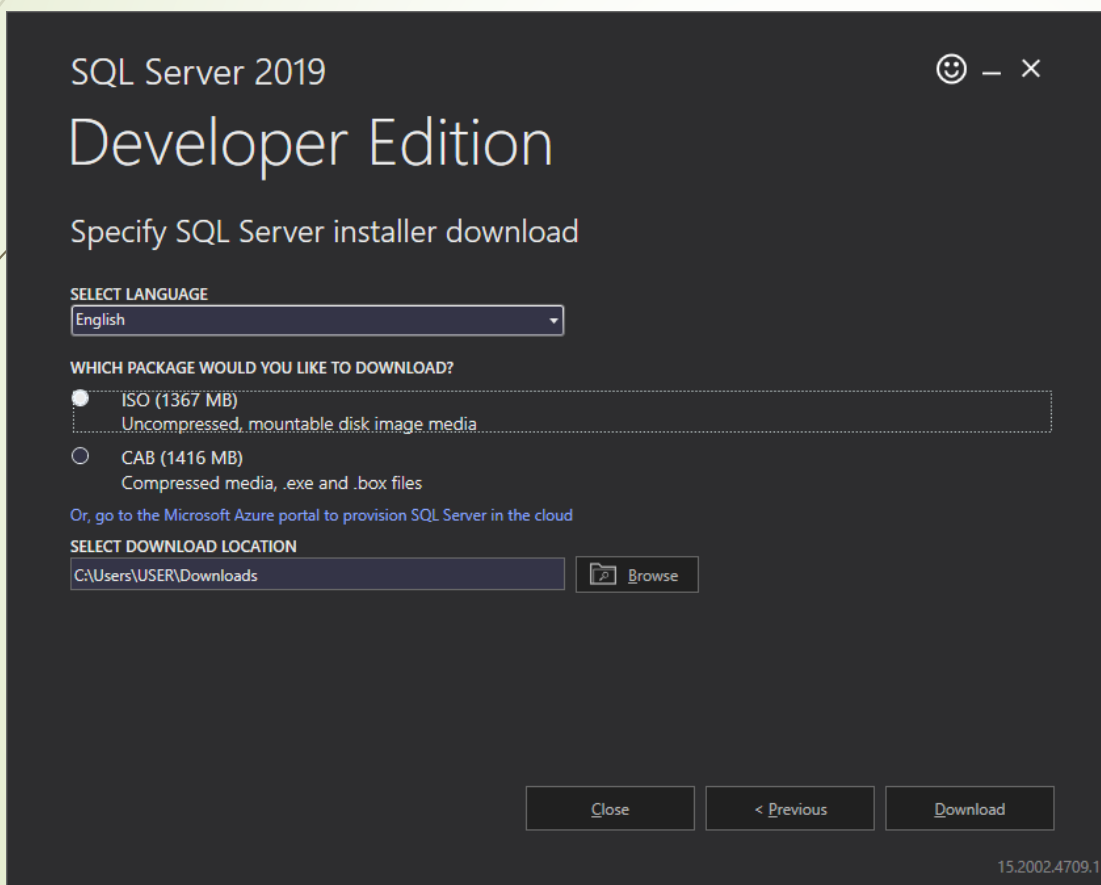


ფაილის გაშვების შემდეგ მონიტორის ეკრანზე გამოისახება ინსტალირების ტიპის შერჩევის ფანჯარა სამი ელემენტით:

- Basic - ჩამოიტვირთება და დაინსტალირდება SQL Server-ის მინიმალური კონფიგურაცია;
- Custom - ჩამოიტვირთვის შემდგომ ინსტალირების ოსტატი დაგეხმარებათ კომპონენტების შერჩევაში და შემდგომ დაინსტალირდება;
- Download Media - ჩამოიტვირთება სრული საინსტალაციო პაკეტი შემდგომ ინსტალირებისათვის.

# MS SQL Server-ის საინსტალაციოს გადმოწერა

ძირითადად ჯობია შერჩეულ იქნეს მესამე ტიპი, რადგან ჩამოტვირთვის შემდგომ მოგეცემა შესაძლებლობა SQL Server-ი დააინსტალიროთ თქვენთვის სასურველ დროს (ასევე მომავალში ინსტალირებული სისტემის აღდგენისას). მესამე ტიპის შერჩევის შემდეგ მომხმარებელს ეძლევა შესაძლებლობა მონიტორის ეკრანზე გამოსახულ ფანჯარაში



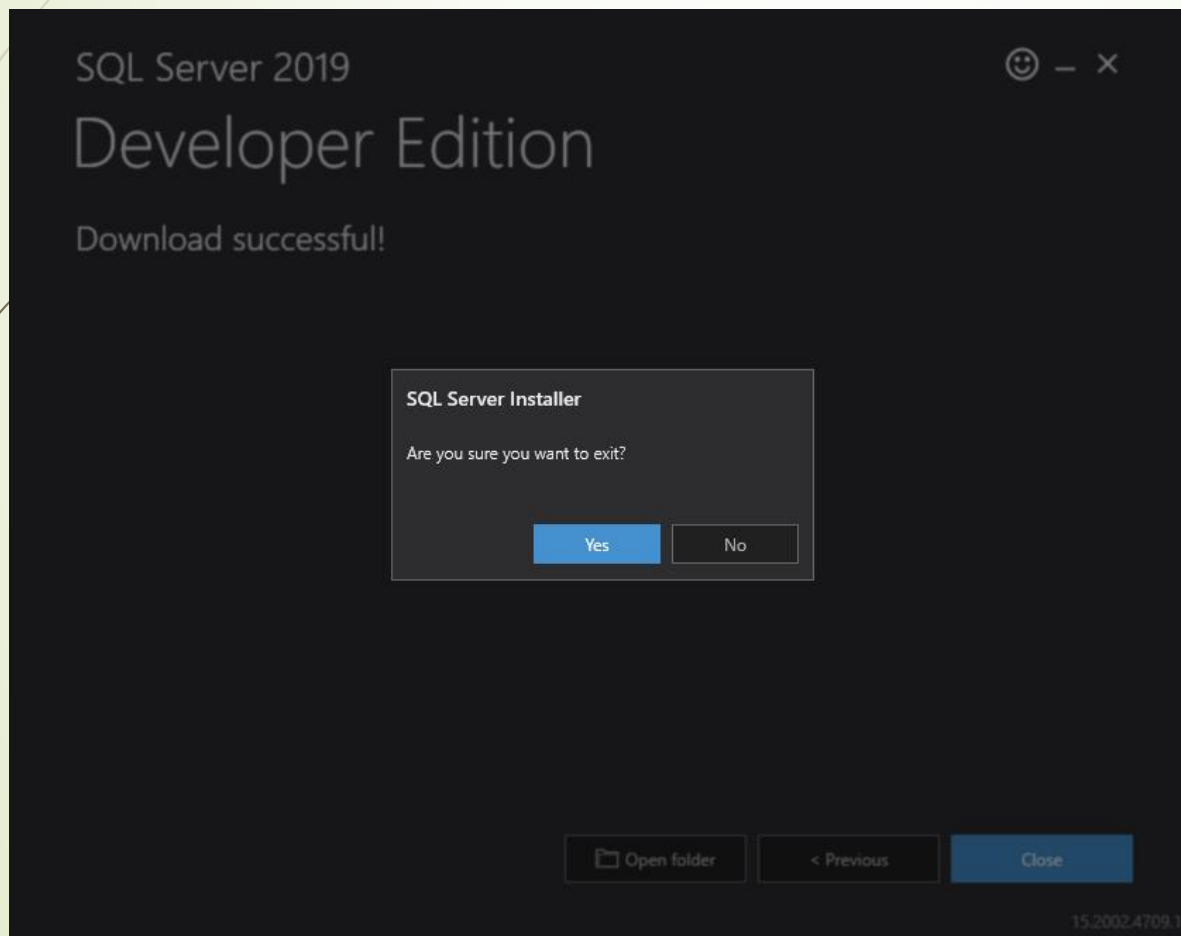
შეარჩიოს ჩამოტვირთვის პარამეტრები:

- SELECT LANGUAGE - SQL Server-ის სიტემის სამუშაო ენა;
- WHICH PACKAGE WOULD YOU LIKE TO DOWNLOAD? - ჩამოტვირთული ფაილების ტიპები (ავტორის რჩევაა - ISO);
- SELECT DOWNLOAD LOCATION - ჩამოსატვირთი ადგილმდებარეობის მითითება.

პარამეტრების შერჩევის შემდგომ უნდა გააქტიურდეს ღილაკი „Download“, რის შემდეგაც ჩამოიტვირთება სისტემა.

# MS SQL Server-ის საინსტალაციოს გადმოწერა

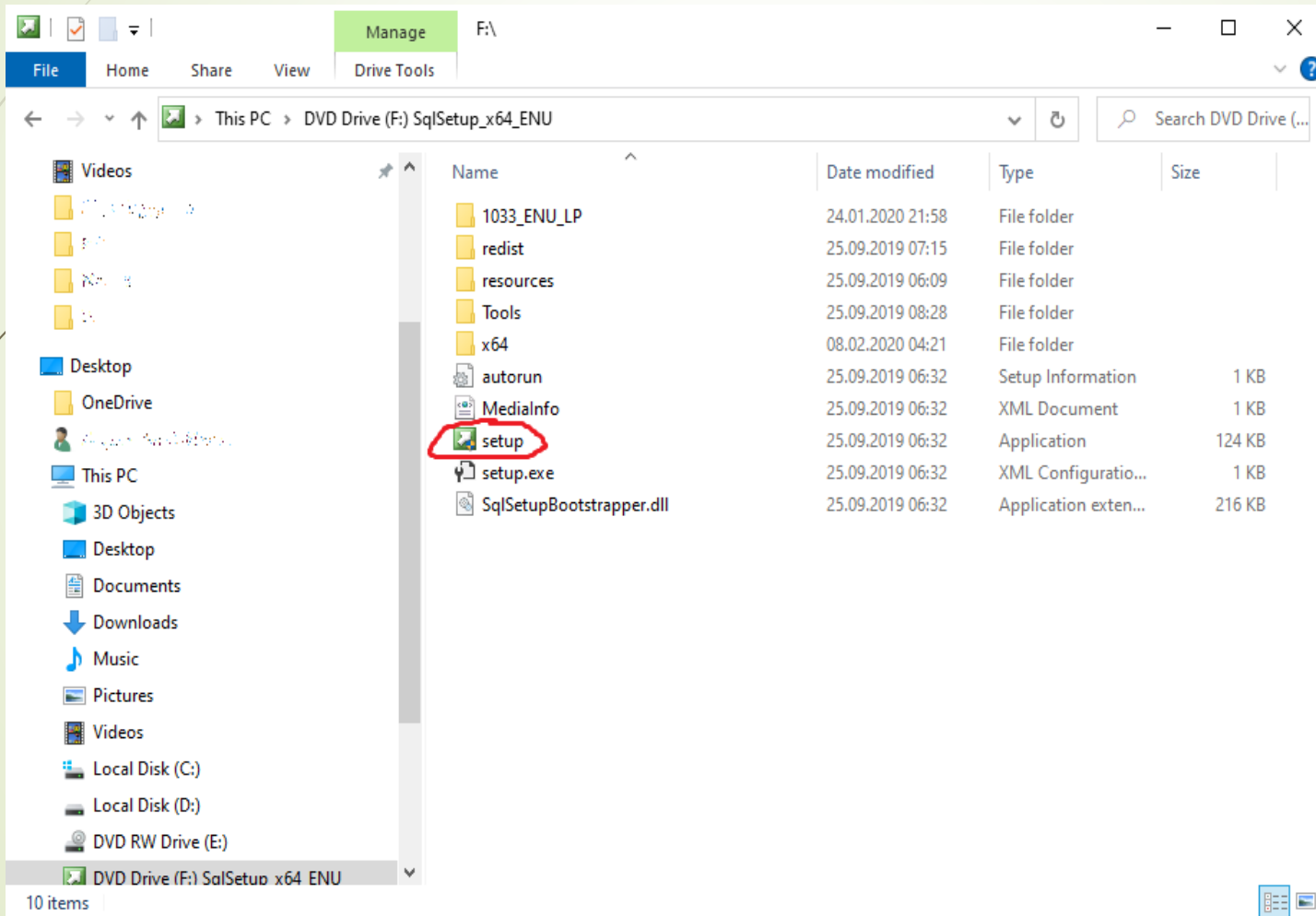
ჩამოტვირთვის დასრულების შემდგომ მონიტორის ეკრანზე გამოისახება დიალოგური ფანჯარა, რომელშიც მომხმარებელს ეძლევა შესაძლებლობა მიუთითოს ინსტალირების დაწყება ან გამოსვლა და დასრულება.



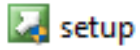


# MS SQL Server-ის ინსტალირების დაგეგმვა

ინსტალირების დაწყებისათვის მომხმარებელმა უნდა მოიძიოს ჩამოტვირთული ფაილი (იმ საქალაქდებში, რომელიც მიუთითა ჩამოტვირთვისას) და გააქტიუროს ფაილი

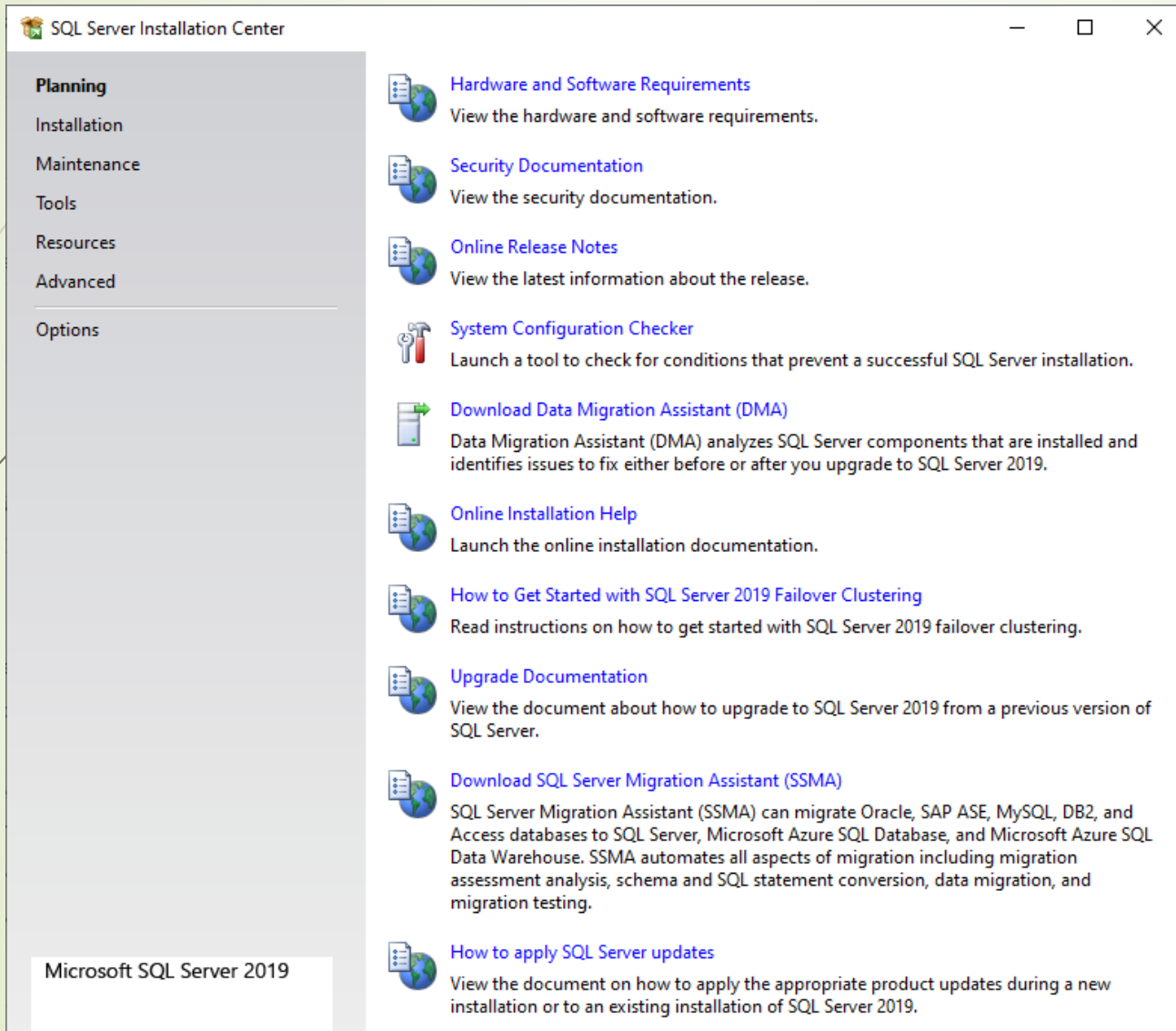


„SqlServer2019-x64-ENU-Dev.iso“.

მონიტორის ეკრანის ფანჯარაში გამოსახება მისი შემცველი  რომელშიც უნდა გააქტიურდეს სტრიქონი (აქაც მომხმარებელს უნდა გააჩნდეს საოპერაციო გარემოში ადმინისტრატორის უფლებები).

# MS SQL Server-ის ინსტალირების დაგეგმვა

7



SQL Server-ის ინსტალირების დაწყების წინ უნდა განხორციელდეს ამ ინსტალირების პროცესის დაგეგმვა.

მონიტორის ეკრანზე ახლად გამოსახულ ფანჯარაში უპირველესყოვლისა ზედა-მარცხენა მიდამოში განთავსებულ სიაში უნდა იქნეს შერჩეული პირველი სტრიქონი Planning, რის შემდეგაც მარჯვენა მიდამოში გამოისახება ინსტალირების დაგეგმვის საშუალებები, დოკუმენტაცია, სისტემის შემოწმება მზადყოფნაზე, მონაცემთა მიგრაციის საშუალებები და სხვა.

# MS SQL Server-ის მინიმალური მოთხოვნები

SQL Server-ის Windows-ში ინსტალირების დაგეგმვისას უნდა იქნეს გათვალისწინებული მინიმალური მოთხოვნები, რომლებიც უნდა დააკმაყოფილოს როგორც კომპიუტერმა, ასევე თვით ოპერაციულმა გარემომ.

SQL Server-ის ინსტალირებისათვის და მუშაობისათვის საჭირო კომპიუტერის მინიმალური რესურსები:

- მონიტორი - SuperVGA (800x600 პიქსელი);
- პროცესორი - აუცილებელი პირობა x64. ტაქტური სიხშირე 1,4 გგჰ (რეკომენდირებულია 2,0 გგჰ);
- ოპერატიული მეხსიერება - 512 მბ Express ვერსიისათვის, ხოლო დანარჩენისათვის 1 გბ (რეკომენდირებულია 1 გბ Express ვერსიისათვის, ხოლო დანარჩენისათვის 4 გბ);
- მყარი დისკი - 6 გბ თავისუფალი სივრცე (აქვე უნდა ითქვას, რომ მყარი დისკი უნდა იყოს სექტორის ზომით 512 ბ ან 4 კბ);
- ინტერნეტი - ფუნქციონალების უზრუნველსაყოფად.



# MS SQL Server-ის მინიმალური მოთხოვნები

SQL Server-ი მხარს უჭერს შემდეგ ქსელურ ოქმებს:

- Shared memory;
- Named Pipes;
- TCP/IP.

SQL Server-ი შეიძლება დაინსტალირდეს ბევრ ნაციონალურ ენაზე, მაგრამ მათ შორის ქართული ჯერ არ არის.

SQL Server-ის ინსტალირებისათვის და მუშაობისათვის მინიმალური ოპერაციული გარემო:

- Windows 10 TH1 1507;
- Windows Server 2016.

უსაფრთხოების მოსაზრებიდან გამომდინარე, არ არის რეკომენდირებული SQL Server-ის ინსტალირება დომენის კონტროლერზე, თუმცა გამოუვალ მდგომარეობაში ინსტალირება შესაძლებელია, მაგრამ შემდეგი შეზღუდვებით:

- სააღრიცხვო ჩანაწერის ლოკალურ სერვისებში SQL Server-ის გაშვება ვერ მოხერხდება;  
(გაგრძელება შემდეგ სლაიდზე)

# MS SQL Server-ის მინიმალური მოთხოვნები

10

- SQL Server-ის ინსტალირების შემდეგ, დომენის წევრი კომპიუტერი ვერ გახდება დომენის კონტროლერი (ამისათვის უნდა განხორციელდეს SQL Server-ის დეინსტალირება);
- SQL Server-ის ინსტალირების შემდეგ, დომენის კონტროლერი კომპიუტერი ვერ გახდება დომენის წევრი (ამისათვის უნდა განხორციელდეს SQL Server-ის დეინსტალირება);
- SQL Server-ი მხარს არ უჭერს მტყუნება მედეგ კლასტერულ ეგზემპლიარებს, რომელშიც კლასტერის კვანძები არიან დომენის კონტროლერები;
- SQL Server-ი დომენის კონტროლერზე მხოლოდ წაკითხვის რეჟიმში ინსტალირებისას, ვერ შექმნის უსაფრთხოების ჯგუფებს, ვერ მოამზადებს სააღრიცხვო ჩანაწერებს და სხვ.

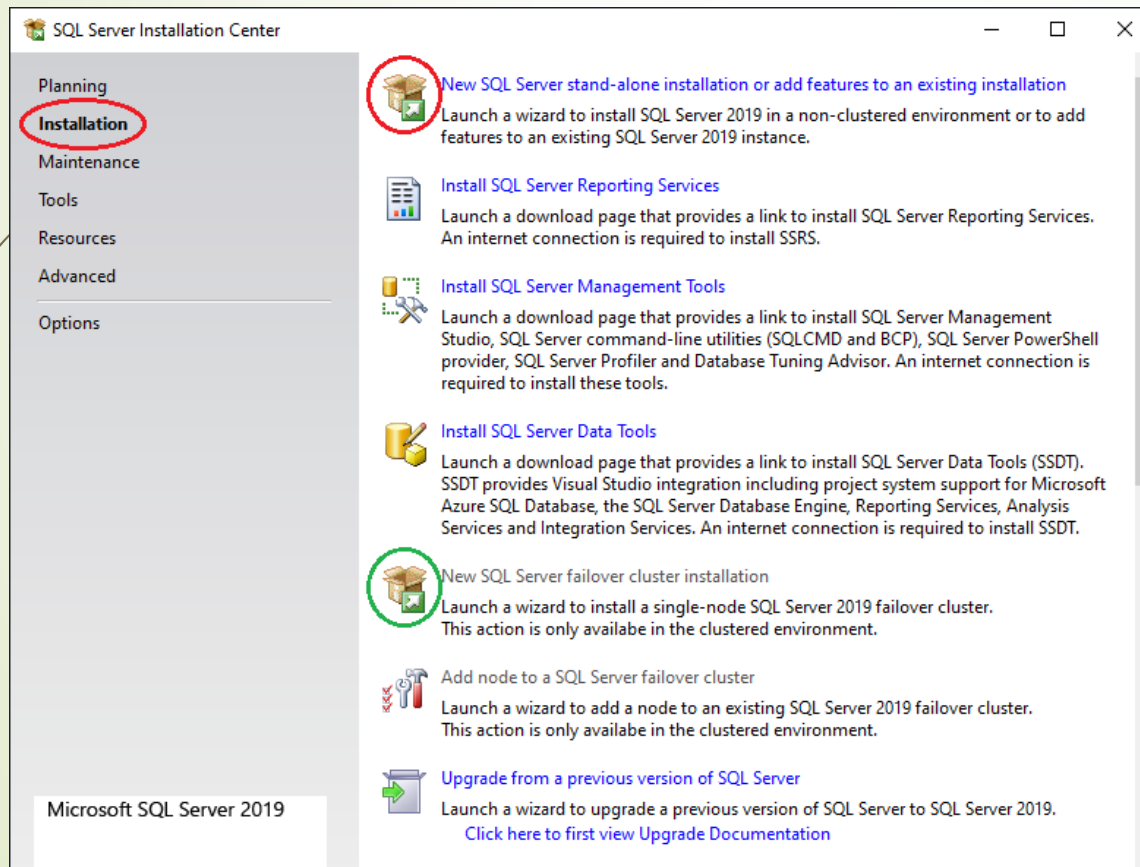
SQL Server-ის ინსტალირება ასევე შესაძლებელია Windows Server 2016 Core და Windows Server 2019 Core-ს ვერსიებშიც (თუმცა ამისათვის მას სჭირდება „.NET Framework 4.6.1“, რომელსაც SQL Server-ი თვითონ დააინსტალირებს გადატვირთვის მოთხოვნით).

ამ ოპერაციულ გარემოებებზე ინსტალირებისას გამოიყენება ავტომატური ინსტალირება (ჩუმი რეჟიმი, ანუ დიალოგური ფანჯრების გარეშე), თუმცა ლიცენზიის პირობებზე თანხმობა მაინს საჭიროა.

# MS SQL Server-ის ინსტალირება Windows-ში

11

SQL Server-ის ინსტალირებისათვის მონიტორის ეკრანზე გამოსახულ დიალოგურ ფანჯარაში ზედა-მარცხენა მიდამოში განთავსებულ სიაში უნდა იქნეს შერჩეული მეორე სტრიქონი Installation, რის შემდგომაც SQL Server-ის ტიპის მიხედვით უნდა შეირჩეს ფანჯრის მარჯვენა მიდამოში გამოსახული სიის მიხედვით. ძირითადებია:



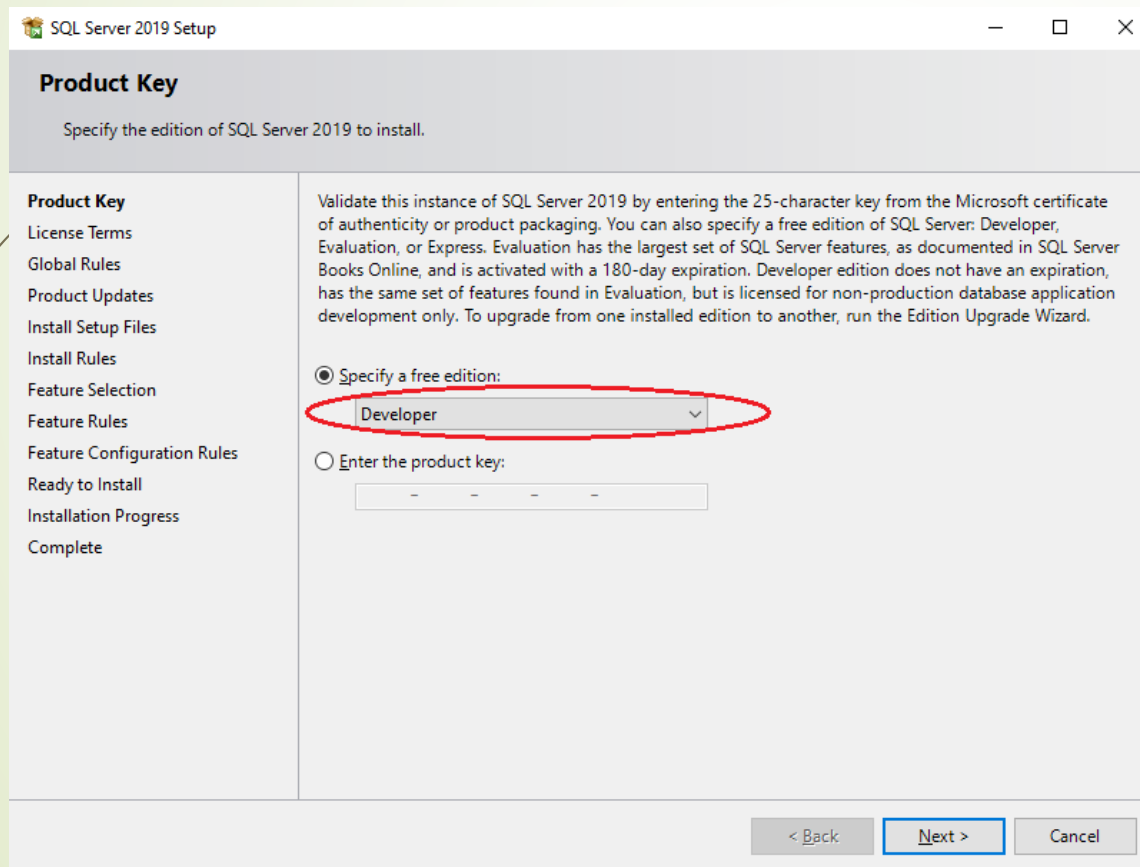
- თუ ინსტალირება ხორციელდება ერთ კომპიუტერზე/სერვერზე, მაშინ უნდა იქნეს შერჩეული პირველი სტრიქონი New SQL Server stand-alone installation or add features to an existing installation;
- თუ ინსტალირება ხორციელდება კლასტერულად (რამდენიმე სერვერზე), ამ შემთხვევაში უნდა იქნეს შერჩეული მეოთხე სტრიქონი New SQL Server cluster installation.



# MS SQL Server-ის ინსტალირება Windows-ში

12

განვიხილოთ ინსტალირება ერთ კომპიუტერზე/სერვერზე, რის შერჩევის შემდეგ დიალოგურ ფანჯარაში მომხმარებელმა უნდა შეიყვანოს (არსებობის შემთხვევაში) პროდუქტის ლიცენზირების გასაღები, ან შეარჩიოს უფასო ვერსია, რომელიც არის სამის ტიპის (უნდა იქნეს შერჩეული - Developer) და გაააქტიუროს ღილაკი Next:



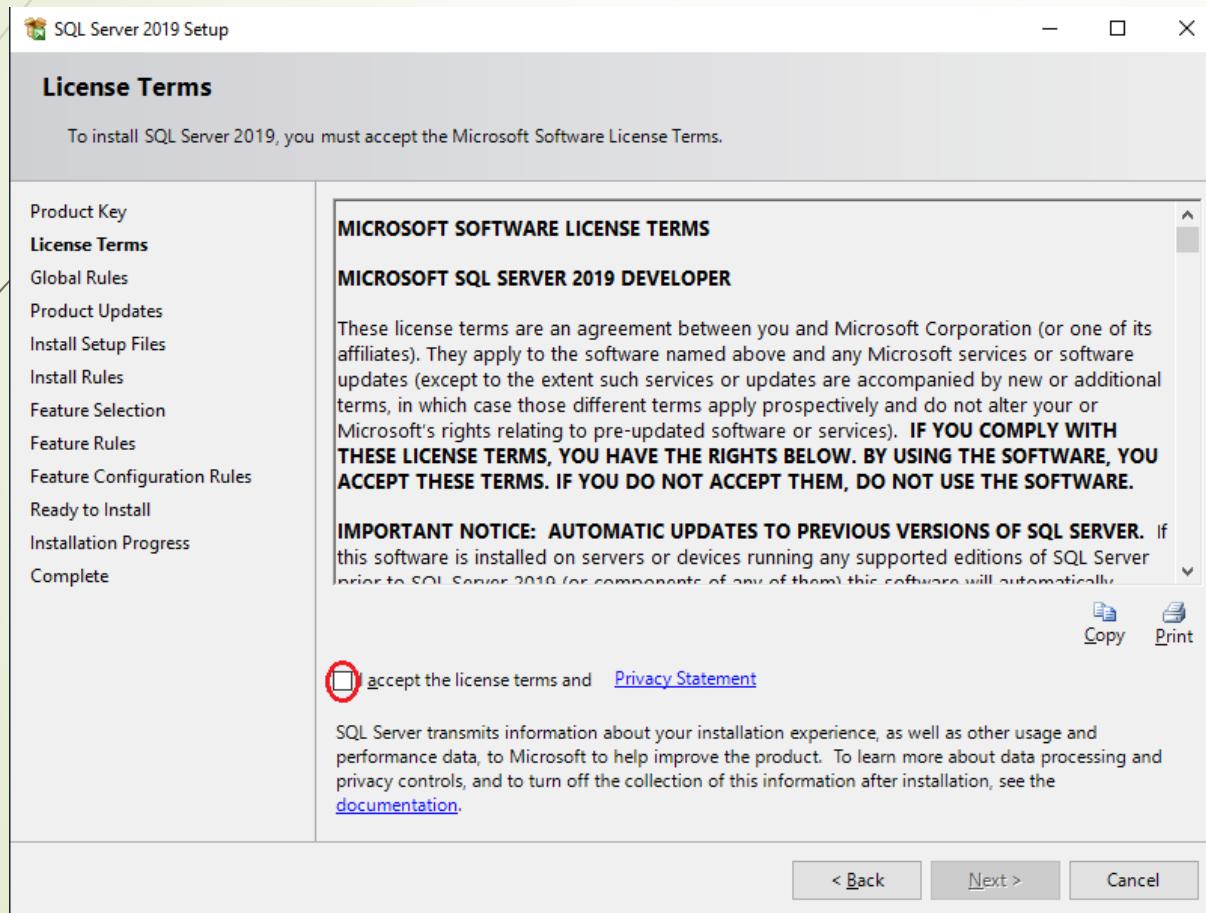
- Evaluation - უფასო ვერსია არის მხოლოდ 180-დღიანი, მაგრამ კომპლექტაციაში მოყვება დოკუმენტაცია, რომელიც მომხმარებელს ეხმარება MS SQL Server-ის ათვისებას;
- Developer - მაქსიმალურად სრული უფასო ვერსიაა, რომლის კომპონენტების გამოსადეგი იქნება დეველოპერებისთვის;
- Express - შეზღუდული უფასო ვერსიაა, რომელშიც შესაძლებელია ელემენტარული ფუნქციების განხორციელება.



# MS SQL Server-ის ინსტალირება Windows-ში

13

შემდგომ დიალოგურ ფანჯარაში მომხმარებელი (გულმოდგინედ წაკითხვისა და გაანალიზების შემდგომ) უნდა დათანხმდეს Microsoft-ის სალიცენზიო პირობებს MICROSOFT SQL SERVER 2019 DEVELOPER პროდუქტთან მიმართებაში.



მომხმარებელს სურვილისებრ აქვს შესაძლებლობა ზემოაღნიშნული პირობები იქვე ქვემო-მარჯვენა ღილაკების გამოყენებით დააკოპიროს ან ამობეჭდოს საბეჭდო მოწყობილობაზე.

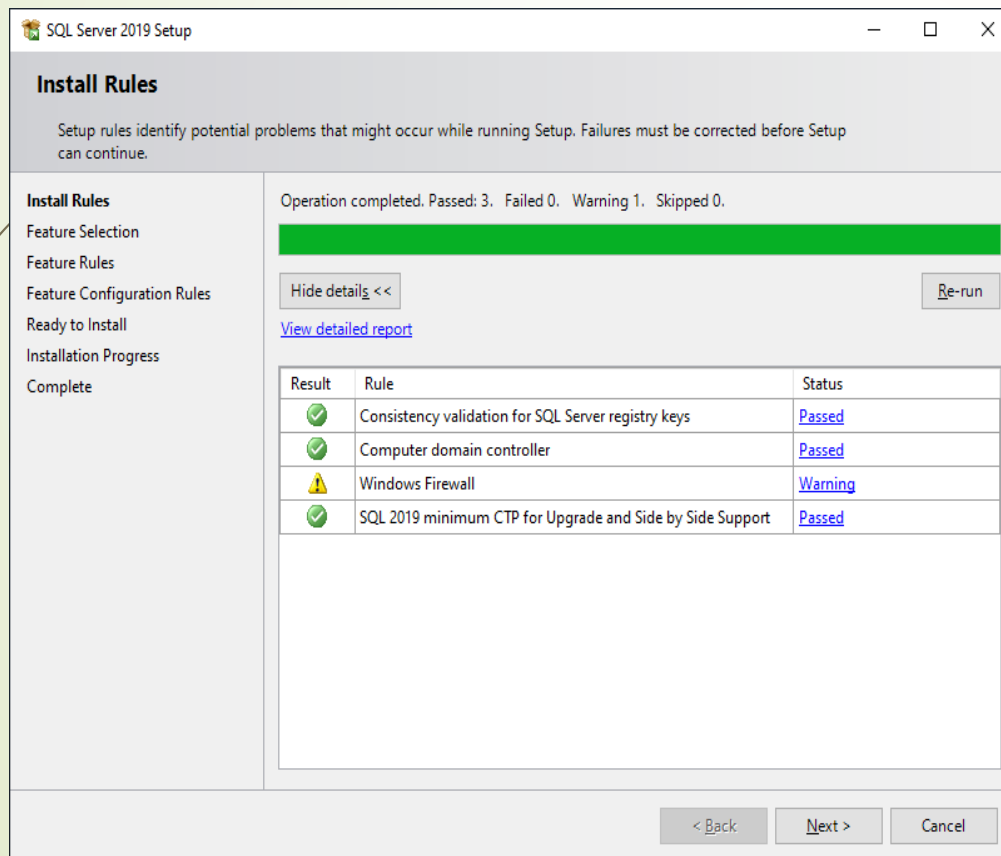
სალიცენზიო პირობებთან დათანხმებისათვის მომხმარებელმა უნდა მონიშნოს დათანხმების ოპცია და გაააქტიუროს ღილაკი Next.

აქვე უნდა აღვნიშნოთ, რომ დათანხმების გარეშე Next ღილაკი არ გააქტიურდება.

# MS SQL Server-ის ინსტალირება Windows-ში

14

შემდგომ დიალოგურ ფანჯარაში გამოისახება ინფორმაცია ინსტალირების პრობლემების შესახებ. იმისდა მიხედვით თუ როგორი შედეგები გამოისახა ფანჯრის შუა მიდამოში მდებარე ცხრილის პირველ სვეტში (Result), შეიძლება მსჯელობა ინსტალირების გაგრძელების შესახებ Next ღილაკის გააქტიურებით:

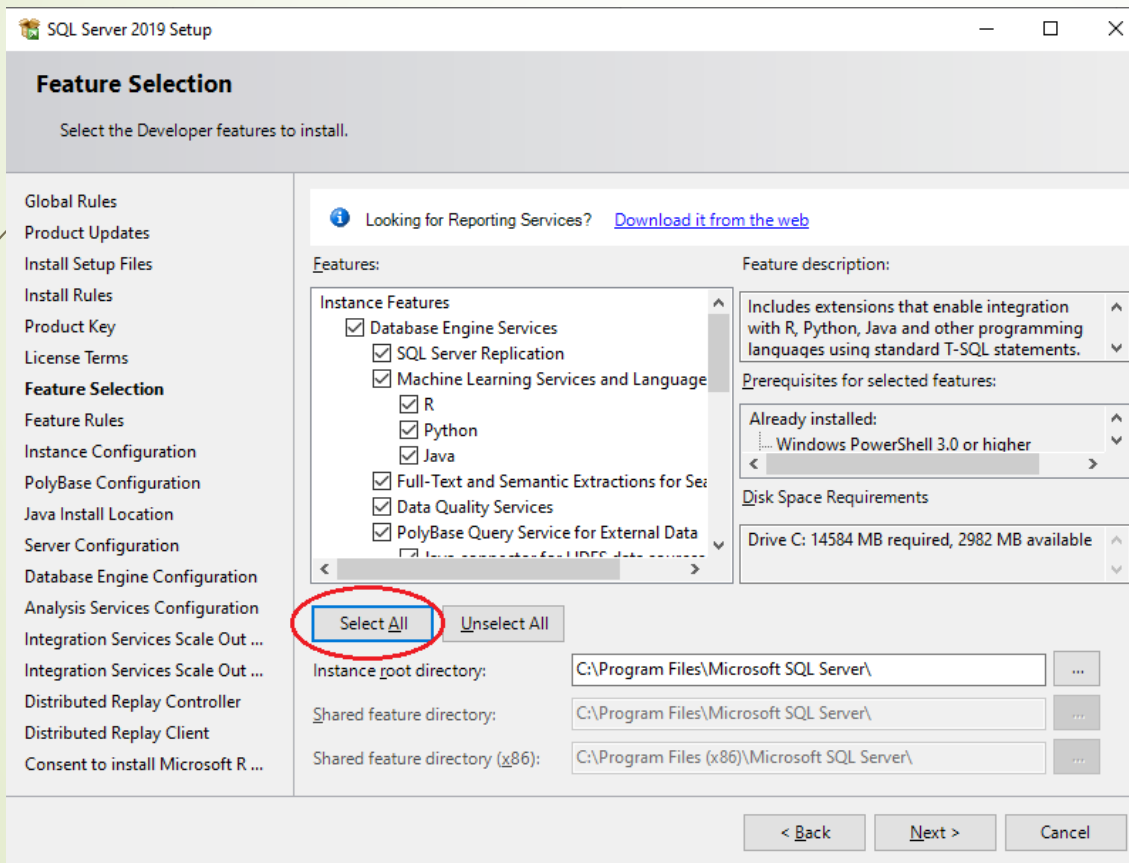


- თუ შედეგის სვეტში მწვანე ხატულ(ებ)ია, ე.ი. მიმდინარე სტრიქონში განთავსებული კომპონენტისათვის ყველაფერი კარგად არის;
- თუ შედეგის სვეტში ყვითელი ხატულ(ებ)ია, ე.ი. მიმდინარე სტრიქონში განთავსებული კომპონენტისათვის რაღაც გასათვალისწინებელია, მაგრამ ინსტალირების გაგრძელება შესაძლებელია;
- თუ შედეგის სვეტში წითელი ხატულ(ებ)ია, ე.ი. მიმდინარე სტრიქონში განთავსებული კომპონენტისათვის არის პრობლემა და ინსტალირება ვერ გაგრძელდება.

# MS SQL Server-ის ინსტალირება Windows-ში

15

შემდგომ დიალოგურ ფანჯარაში მომხმარებელმა ფანჯრის შუა მიდამოში გამოსახულ სიაში უნდა მონიშნოს თუ რომელი პარამეტრების ინსტალირება არის საჭირო. ამოცანებიდან გამომდინარე მომხმარებელს შეუძლია ღილაკით Select All მონიშნოს ყველა პუნქტი გარდა:



➤ მანქანური სწავლების ჯგუფებისა (Machine Learning Services and Language Extensions და Machine Learning Server(Standalone)), რომელშიც გათვალისწინებულ იქნება R, Python, Java და სხვა პროგრამული ენების ინტეგრაცია T-SQL-ში;

➤ PolyBase მოთხოვნების სერვისის ჯგუფის (PolyBase Query Service for External for External Data).

შემდეგ ფანჯარაზე გადასასვლელად უნდა გააქტიურდეს ღილაკი Next.



# MS SQL Server-ის ინსტალირება Windows-ში

16

წინა დიალოგურ ფანჯარაში მოსანიშნი კომპონენტების დანიშნულებებია:

- Database Engine SQL Server - მონაცემთა ბაზის კომპონენტის ინსტალირება;
- SQL Server Replication - არააუცილებელი კომპონენტი მონაცემთა ბაზებს შორის მონაცემთა და მონაცემთა ბაზის ობიექტების კოპირებისა და გავრცელების ტექნოლოგიების ერთობლიობა და ასევე მონაცემთა ბაზების სინქრონიზაციისათვის;
- Machine Learning Services and Language - არააუცილებელი კომპონენტი მანქანური სწავლების სერვისების (R და Python) და ენის გაფართოებისათვის (Java);
- Full-Text and Semantic Extractions for Search - არააუცილებელი კომპონენტი სრულტექსტოვანი და სემანტიკური გაფართოების ძეგლისათვის;
- Data Quality Services (DQS) - არააუცილებელი კომპონენტი მონაცემთა ხარისხის სერვისებისათვის (DQS-ის ინსტალირებისათვის SQL Server-ის ინსტალირების დასრულების შემდეგ საჭიროა დამატებითი მოქმედებები);
- Polybase Query Service for External Data - არააუცილებელი კომპონენტი Java-ს HDFS მონაცემთა წყაროსთან დასაკავშირებლად.

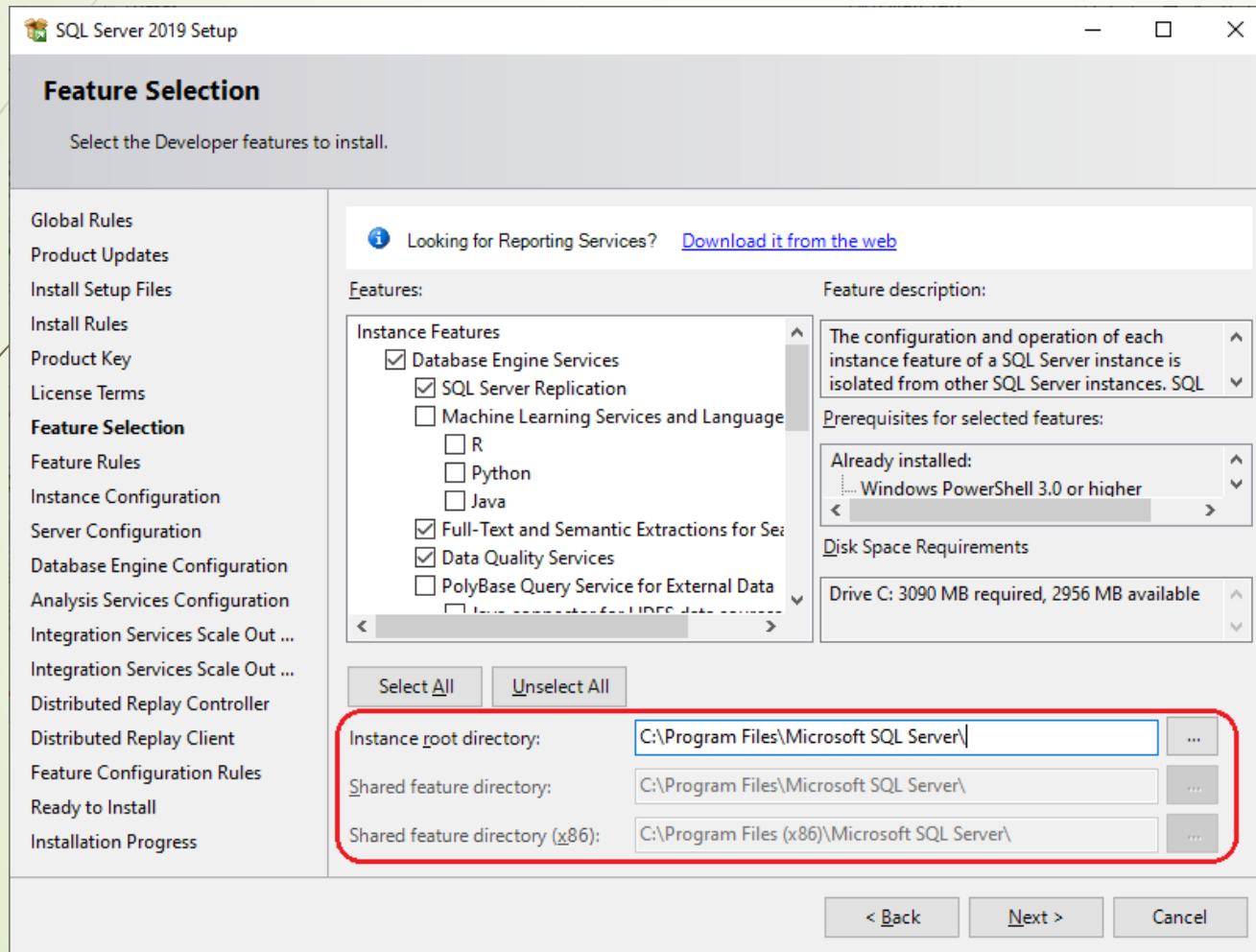
დანარჩენი დამატებითი შესაძლებლობები შესაძლებელია გამოყენებულ იქნეს მრავალ ტიპიურ მომხმარებლის სცენარებში.



# MS SQL Server-ის ინსტალირება Windows-ში

17

ამავე დიალოგურ ფანჯარაში მომხმარებელს ეძლევა შესაძლებლობა შეცვალოს MS SQL Server-ის სტანდარტული საინსტალაციო გზები:



შემდეგ ფანჯარაზე გადასასვლელად უნდა გააქტიურდეს ლილაკი Next.

# MS SQL Server-ის ინსტალირება Windows-ში

შემდგომ დიალოგურ ფანჯარაში მომხმარებელს ეძლევა შესაძლებლობა დააფიქსიროს MS SQL Server-ის სახელი:

- Default Instance - სტანდარტული სახელი MSSQLSERVER;
- Named instance - ნებისმიერი სახელი, რომელიც სურს მომხმარებელს, მხოლოდ უნდა იქნეს დაცული სისტემის მიერ დასახული სახელების წესები:

**Instance Configuration**

Specify the name and instance ID for the instance of SQL Server. Instance ID becomes part of the installation path.

Global Rules  
Product Updates  
Install Setup Files  
Install Rules  
Product Key  
License Terms  
Feature Selection  
Feature Rules  
**Instance Configuration**  
Server Configuration  
Database Engine Configuration  
Analysis Services Configuration  
Integration Services Scale Out ...  
Integration Services Scale Out ...  
Distributed Replay Controller  
Distributed Replay Client  
Feature Configuration Rules  
Ready to Install  
Installation Progress

☒ Default instance  
☐ Named instance:

Instance ID:

SQL Server directory: D:\SQL19\MSSQL15.MSSQLSERVER  
Analysis Services directory: D:\SQL19\MSAS15.MSSQLSERVER

Installed instances:

Instance Name	Instance ID	Features	Edition	Version
NA	MSSQL14.NA,MSA...	SQLEngine,SQLEn...	Developer	14.0.2037.2
<Shared Compone...		Conn, BC, SDK, IS\...		14.0.1000.169
<Shared Compone...		DQC, IS, MDS		14.0.2037.2

< Back Next > Cancel

- ვერ დაიწყება და ვერ დასრულდება ქვეშ ხაზგასმის „\_“ სიმბოლოთი;
- ვერ დაერქმევა დარეზერვებული სიტყვა (მაგ. Default);
- ზომა 16 სიმბოლომდე;
- პირველი სიმბოლო Unicode 2.0;
- მეორე სიმბოლოდან Unicode 2.0, ათწილადი რიცხვები, დოლარის ნიშანი „\$“ და ქვეშ ხაზგასმის „\_“ სიმბოლო.

შემდეგ ფანჯარაზე გადასასვლელად უნდა გააქტიურდეს ლილაკი Next.

# MS SQL Server-ის ინსტალირება Windows-ში

19

შემდგომ დიალოგურ ფანჯრის პირველ ჩანართში Service Accounts მომხმარებელს ეძლევა შესაძლებლობა MS SQL Server-ის ყველა სერვისებისათვის დააფიქსიროს მოწყვლადობის პაროლები, ხოლო მეორე ჩანართში Collation - მიუთითოს სისტემას მონაცემთა ბაზის და ანალიტიკური სერვისების მონაცემთა ენის კოდირებები:

SQL Server 2019 Setup

**Server Configuration**  
Specify the service accounts and collation configuration.

Global Rules  
Product Updates  
Install Setup Files  
Install Rules  
Product Key  
License Terms  
Feature Selection  
Feature Rules  
Instance Configuration  
**Server Configuration**  
Database Engine Configuration  
Analysis Services Configuration  
Integration Services Scale Out ...  
Integration Services Scale Out ...  
Distributed Replay Controller  
Distributed Replay Client  
Feature Configuration Rules  
Ready to Install  
Installation Progress

Service Accounts Collation

Microsoft recommends that you use a separate account for each SQL Server service.

Service	Account Name	Password	Startup Type
SQL Server Agent	NT Service\SQLSERVERA...		Manual
SQL Server Database Engine	NT Service\MSSQLSERVER		Automatic
SQL Server Analysis Services	NT Service\MSSQLServer...		Automatic
SQL Server Integration Services 15.0	NT Service\MsDtsServer...		Automatic
SQL Server Integration Services Sc...	NT Service\SSISScaleOut...		Automatic
SQL Server Integration Services Sc...	NT Service\SSISScaleOut...		Automatic
SQL Server Distributed Replay Client	NT Service\SQL Server Di...		Manual
SQL Server Distributed Replay Con...	NT Service\SQL Server Di...		Manual
SQL Full-text Filter Daemon Launc...	NT Service\MSSQLFDLa...		Manual
SQL Server Browser	NT AUTHORITY\LOCALS...		Automatic

☐ Grant Perform Volume Maintenance Task privilege to SQL Server Database Engine Service  
This privilege enables instant file initialization by avoiding zeroing of data pages. This may lead to information disclosure by allowing deleted content to be accessed.

< Back Next > Cancel

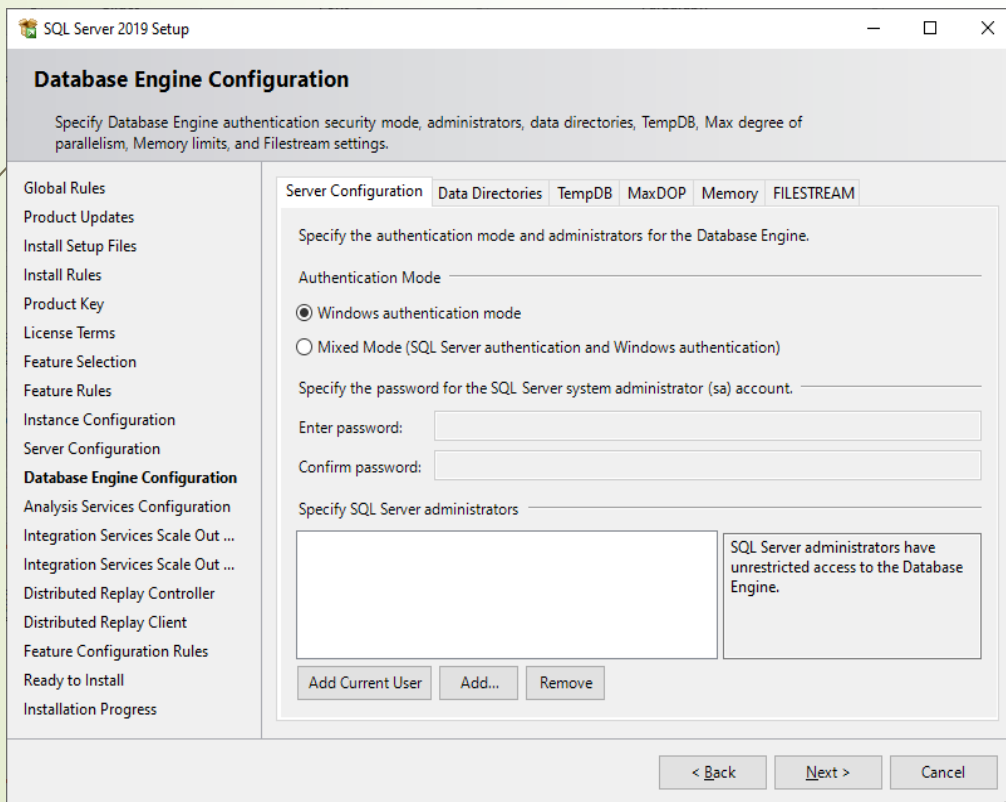
შემდეგ ფანჯარაზე გადასასვლელად უნდა გააქტიურდეს ღილაკი Next.



# MS SQL Server-ის ინსტალირება Windows-ში

შემდგომი დიალოგური ფანჯრის პირველ ჩანართში Server Configuration მომხმარებელს ეძლევა შესაძლებლობა განსაზღვროს MS SQL Server-ის ავთენტიფიკაციის რეჟიმი:

- Windows authorization mode – MS SQL Server-ში ავთენტიფიკაცია განხორციელდება Windows-ის ავთენტიფიკაციის მიხედვით (ეს ვარიანტი უფრო დაცულია, რადგან უსაფრთხოება კონტროლირდება Windows-ის უსაფრთხოების ოქმით Kerberos);



- Mixed Mode (SQL Server authentication and Windows authentication) – შერეული რეჟიმი, ანუ MS SQL Server-ში ავთენტიფიკაცია განხორციელდება როგორც Windows-ის, ასევე SQL Server-ის ავთენტიფიკაციით, რომლის პაროლიც უნდა მიეთითოს ამავე ფანჯარაში (128-მდე ასო, ციფრი და სიმბოლოები).

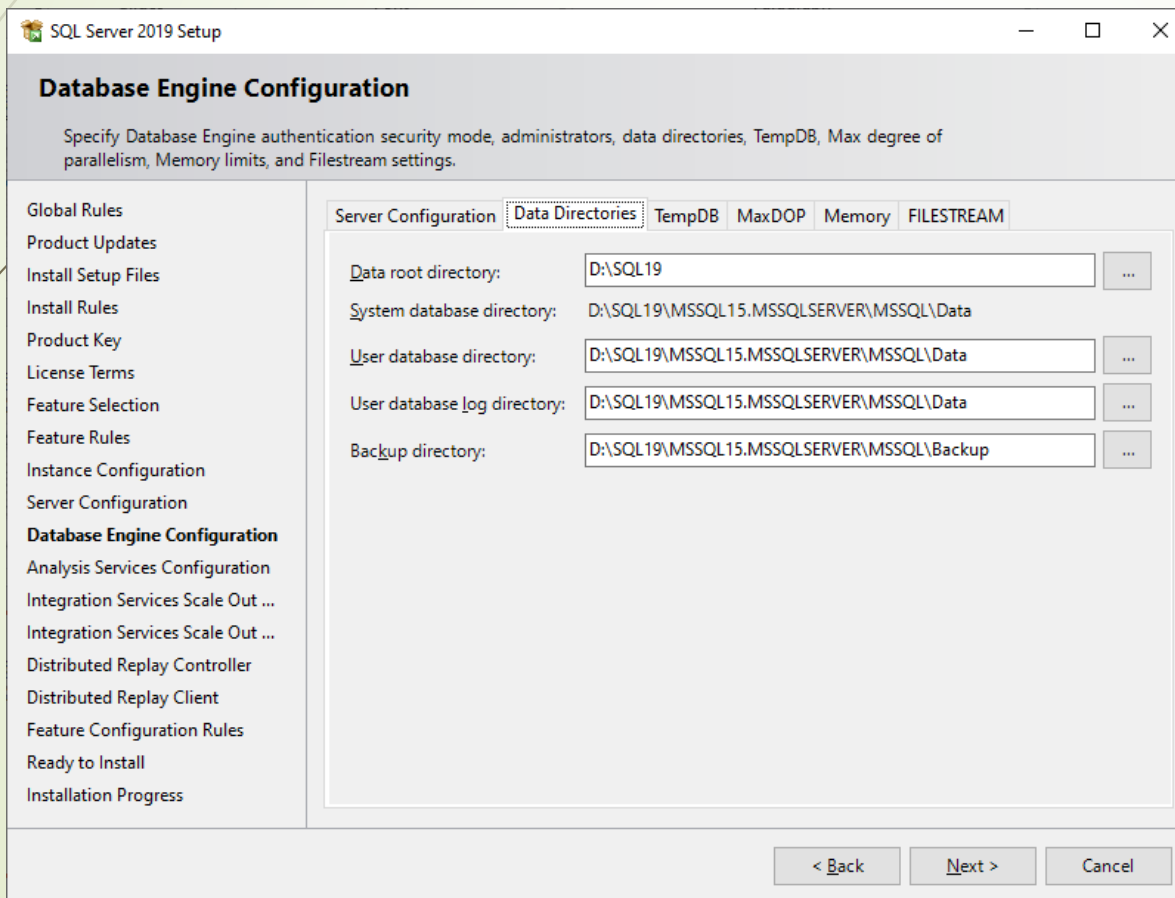
ასევე, ფანჯრის ქვედა მიდამოში აუცილებლად უნდა იქნენ მითითებულნი SQL Server-ის ადმინისტრატორები (მაგალითად, ღილაკის Add Current User გააქტიურებით დაემატება მიმდინარე მომხმარებელი).



# MS SQL Server-ის ინსტალირება Windows-ში

იგივე დიალოგური ფანჯრის მეორე ჩანართში Server Configuration მომხმარებელს ეძლევა შესაძლებლობა განსაზღვროს MS SQL Server-ის მონაცემთა საქალაქდები:

➤ Data root directory – მონაცემთა ძირითადი საქალაქდე;



➤ System database directory – სისტემური მონაცემთა ბაზების საქალაქდე;

➤ User database directory – მომხმარებლის მონაცემთა ბაზების საქალაქდე;

➤ User database log directory – მომხმარებლის მონაცემთა ბაზების ჟურნალების საქალაქდე;

➤ Backup directory – მონაცემთა ბაზების სარეზერვო ასლების საქალაქდე.

# MS SQL Server-ის ინსტალირება Windows-ში

იგივე დიალოგური ფანჯრის მესამე ჩანართში TempDB მომხმარებელს ეძლევა შესაძლებლობა განსაზღვროს MS SQL Server-ის დროებითი მონაცემთა ბაზების რეკვიზიტები: ზედა მიდამოში დროებითი მონაცემთა ბაზების ფაილების (TempDB data files) და ქვედა მიდამოში დროებითი მონაცემთა ბაზების ჟურნალის ფაილების (TempDB log file):

The screenshot shows the 'Database Engine Configuration' window for SQL Server 2019 Setup, specifically the 'TempDB' tab. The window is divided into two main sections: a left sidebar with navigation links and a main configuration area on the right.

**Left Sidebar (Navigation):**

- Global Rules
- Product Updates
- Install Setup Files
- Install Rules
- Product Key
- License Terms
- Feature Selection
- Feature Rules
- Instance Configuration
- Server Configuration
- Database Engine Configuration** (Selected)
- Analysis Services Configuration
- Integration Services Scale Out ...
- Integration Services Scale Out ...
- Distributed Replay Controller
- Distributed Replay Client
- Feature Configuration Rules
- Ready to Install
- Installation Progress

**Main Configuration Area (TempDB Tab):**

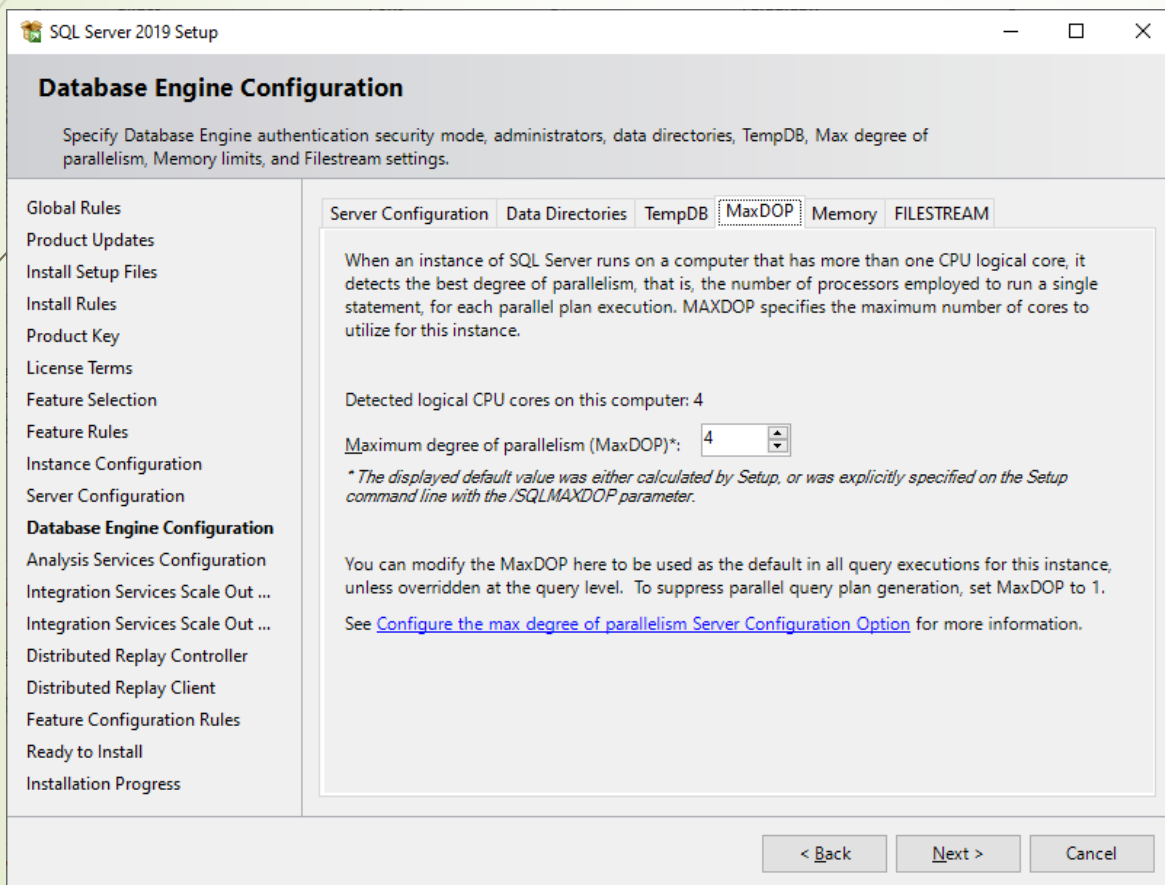
- TempDB data files:** tempdb.mdf, tempdb\_mssql\_#.ndf
- Number of files:** 4 (spin box)
- Initial size (MB):** 8 (spin box) | **Total initial size (MB):** 32 (text box)
- Autogrowth (MB):** 64 (spin box) | **Total autogrowth (MB):** 256 (text box)
- Data directories:** D:\SQL19\MSSQL15.MSSQLSERVER\MSSQL\Data (text box with 'Add...' and 'Remove' buttons)
- TempDB log file:** templog.ldf
- Initial size (MB):** 8 (spin box) | Setup could take longer with large initial size.
- Autogrowth (MB):** 64 (spin box)
- Log directory:** D:\SQL19\MSSQL15.MSSQLSERVER\MSSQL\Data (text box with '...' button)

**Bottom Buttons:** < Back, Next >, Cancel

- Number of files – ფაილების რაოდენობა;
- Initial size – საწყისი ზომა;
- Autogrowth – ნაზრდის ზომა;
- Data directories – დროებითი მონაცემთა ბაზების საქაღალდე;
- Log directory – მონაცემთა ბაზების ჟურნალის საქაღალდე.

# MS SQL Server-ის ინსტალირება Windows-ში

იგივე დიალოგური ფანჯრის მეოთხე ჩანართში MaxDOP მომხმარებელს ეძლევა შესაძლებლობა განსაზღვროს მიმდინარე კომპიუტერის პროცესორის ბირთვების რაოდენობის მიხედვით MS SQL Server-ის ამოცანების გადაწყვეტის პარალელიზმის ხარისხი (რაოდენობა):



# MS SQL Server-ის ინსტალირება Windows-ში

იგივე დიალოგური ფანჯრის მეხუთე ჩანართში Memory მომხმარებელს ეძლევა შესაძლებლობა განსაზღვროს MS SQL Server-ის მუშაობისათვის საჭირო კომპიუტერის მეხსიერების რეკომენდირებული ან სტანდარტული მინიმალური და მაქსიმალური ზომა:

**SQL Server 2019 Setup**

**Database Engine Configuration**

Specify Database Engine authentication security mode, administrators, data directories, TempDB, Max degree of parallelism, Memory limits, and Filestream settings.

Global Rules  
Product Updates  
Install Setup Files  
Install Rules  
Product Key  
License Terms  
Feature Selection  
Feature Rules  
Instance Configuration  
Server Configuration  
**Database Engine Configuration**  
Analysis Services Configuration  
Integration Services Scale Out ...  
Integration Services Scale Out ...  
Distributed Replay Controller  
Distributed Replay Client  
Feature Configuration Rules  
Ready to Install  
Installation Progress

Server Configuration | Data Directories | TempDB | MaxDOP | **Memory** | FILESTREAM

SQL Server can change its memory requirements dynamically based on available system resources. However, in some scenarios you can configure the range of memory (in MB) that is managed by the SQL Server Memory Manager for this instance, by specifying min server memory and/or max server memory.

☐ Recommended ☒ **Default**

Min Server Memory (MB): 0 0

Max Server Memory (MB): 5940 2147483647

*\* The displayed recommended values were calculated by Setup based on your system configuration and edition, unless these were explicitly specified in the Setup command line using the /SQLMINMEMORY and /SQLMAXMEMORY parameters.*

For more information see: [Server Memory Server Configuration Options](#).

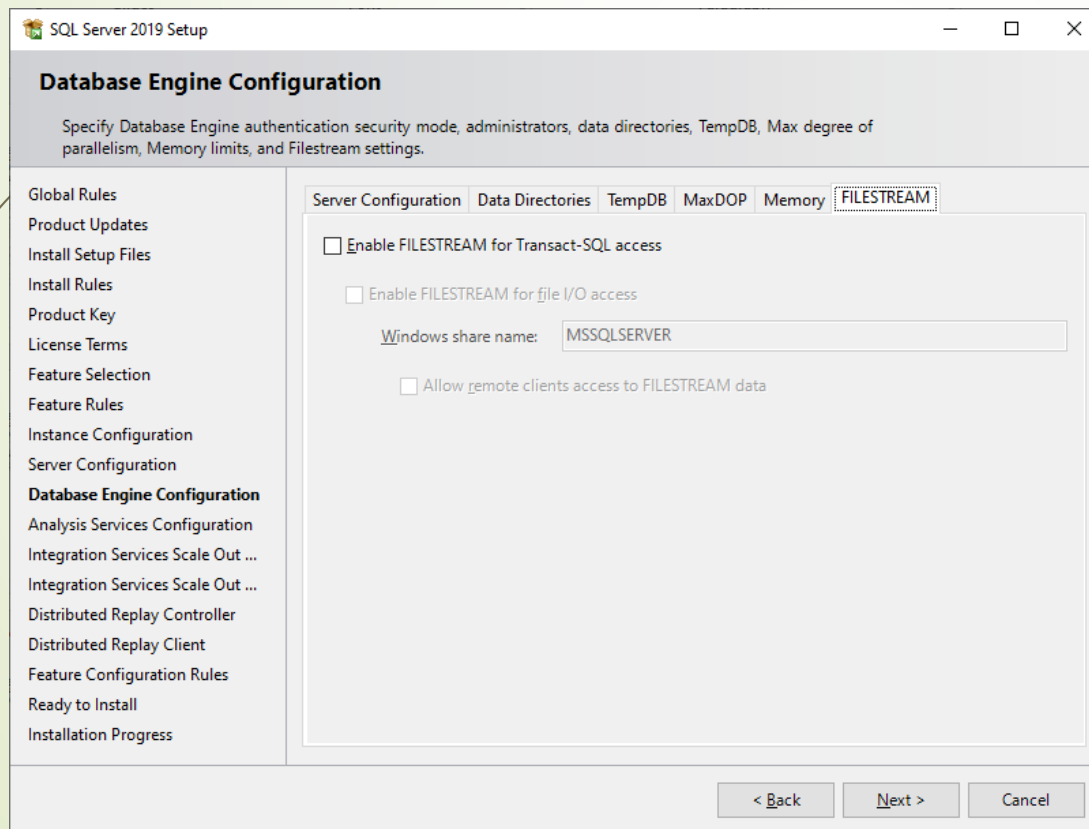
☐ Click here to accept the recommended memory configurations for the SQL Server Database Engine

< Back Next > Cancel



# MS SQL Server-ის ინსტალირება Windows-ში

იგივე დიალოგური ფანჯრის მეექვსე ჩანართში FILESTREAM მომხმარებელს ეძლევა შესაძლებლობა განსაზღვროს MS SQL Server-ის მონაცემთა ბაზის კომპონენტის Database Engine-ს გამოყენებით შეინახოს NTFS ფაილურ სისტემაში varbinary(max) ტიპის დიდი მონაცემების ორობითი ობიექტები (BLOB) ფაილების სახით, რომლის ჩასართავად უნდა მოინიშნოს Enable FILESTREAM for Transact-SQL Access.



Win32-ს ნაკადური წვდომის დაშვებისათვის უნდა მოინიშნოს Enable FILESTREAM for file I/O access.

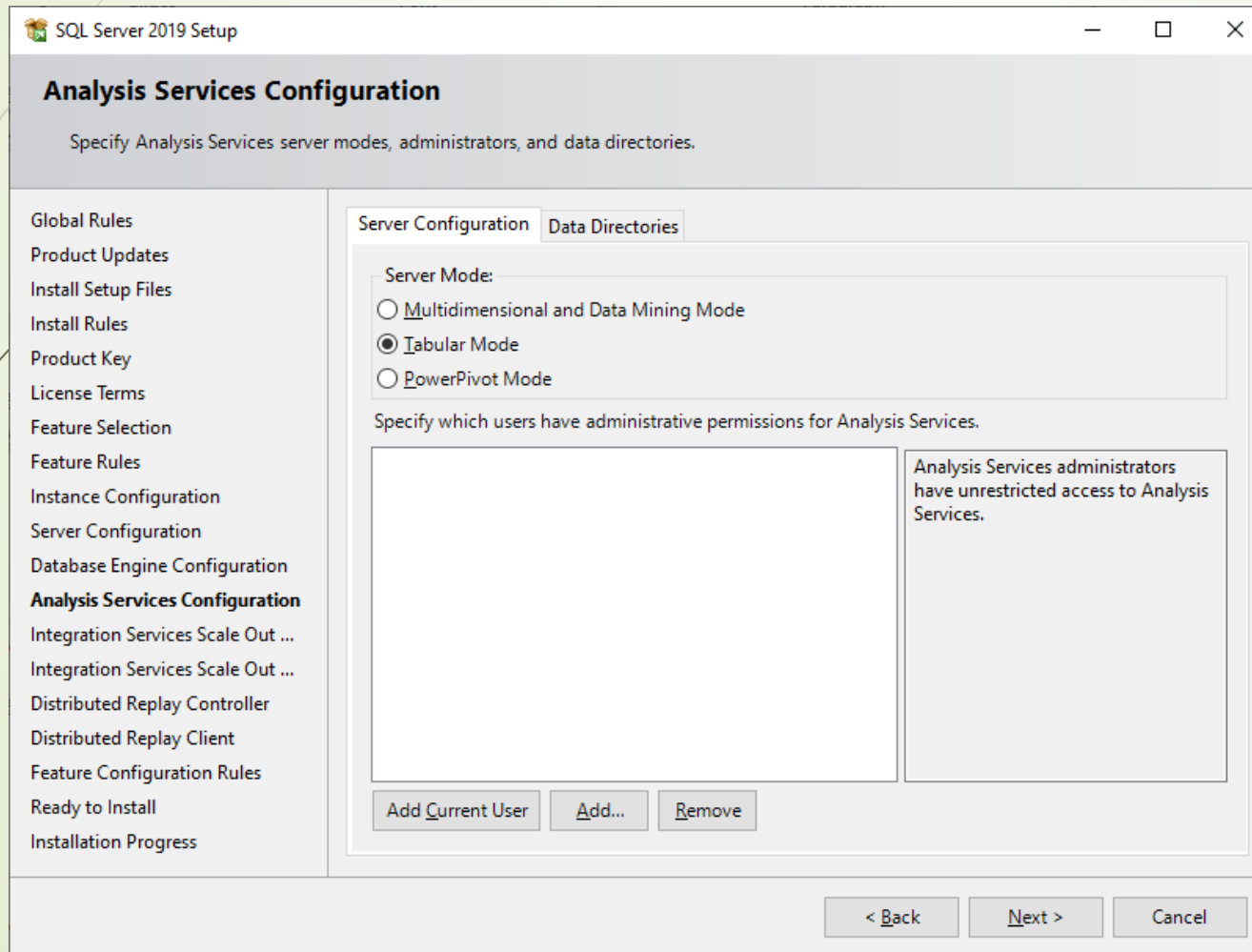
FILESTREAM-ის მონაცემთა შენახვის საქალაქის სახელი უნდა ჩაიწეროს Windows share name სტრიქონში.

ამ ფოლდერზე დაშორებული კლიენტების წვდომისათვის უნდა მოინიშნოს Allow remote clients access for FILESTREAM data.

შემდეგ ფანჯარაზე გადასასვლელად უნდა გააქტიურდეს ღილაკი Next.

# MS SQL Server-ის ინსტალირება Windows-ში

შემდგომ დიალოგურ ფანჯარაში მომხმარებელს ეძლევა შესაძლებლობა დააფიქსიროს MS SQL Server-ის ანალიტიკური სერვისების კონფიგურირების რეჟიმები, ხოლო ფანჯრის



ქვედა მიდამოში უნდა იქნენ მითითებულნი SQL Server-ის ანალიტიკური სერვისების ადმინისტრატორები (მაგალითად, ღილაკის Add Current User გააქტიურებით დაემატება მიმდინარე მომხმარებელი).

შემდეგ ფანჯარაზე გადასასვლელად უნდა გააქტიურდეს ღილაკი Next.

# MS SQL Server-ის ინსტალირება Windows-ში

შემდგომ დიალოგურ ფანჯარაში მომხმარებელს ეძლევა შესაძლებლობა დააფიქსიროს MS SQL Server-ის სამუშაო პორტის ნომერი (სტანდარტულად იგი არის 8391), შექმნას ან მიუთითოს არსებული უსაფრთხოების SSL-სერტიფიკატი:

SQL Server 2019 Setup

### Integration Services Scale Out Configuration - Master Node

Specify the port number and security certificate for the Scale Out Master node.

Global Rules  
Product Updates  
Install Setup Files  
Install Rules  
Product Key  
License Terms  
Feature Selection  
Feature Rules  
Instance Configuration  
Server Configuration  
Database Engine Configuration  
Analysis Services Configuration  
**Integration Services Scale Ou...**  
Integration Services Scale Out ...  
Distributed Replay Controller  
Distributed Replay Client  
Feature Configuration Rules  
Ready to Install  
Installation Progress

Specify a port number that the master node uses to communicate with the worker nodes.

Port Number:

Select a SSL certificate that is used for the communication between the master node and worker nodes in the scale out topology. A default self-signed certificate is created if you choose to create a new SSL certificate.

☒ Create a new SSL certificate

CNs in the certificate:

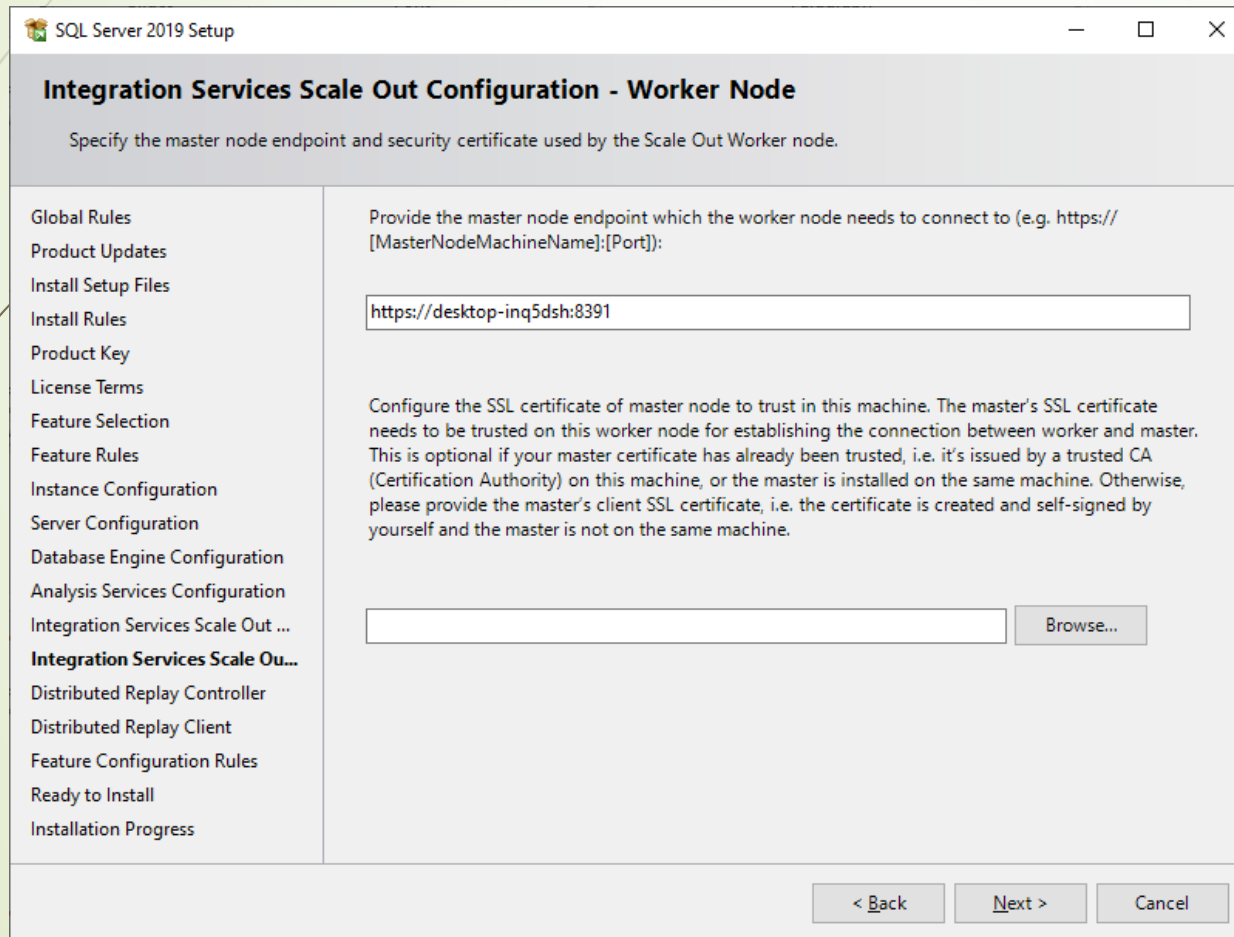
☐ Use an existing SSL certificate

< Back   Next >   Cancel

შემდეგ ფანჯარაზე გადასასვლელად უნდა გააქტიურდეს ღილაკი Next.

# MS SQL Server-ის ინსტალირება Windows-ში

შემდგომ დიალოგურ ფანჯარაში მომხმარებელს ეძლევა შესაძლებლობა დააფიქსიროს MS SQL Server-თან მიერთების ვებ-მისამართი და რეჟიმის ოსტატის კონფიგურირების SSL-სერტიფიკატი:



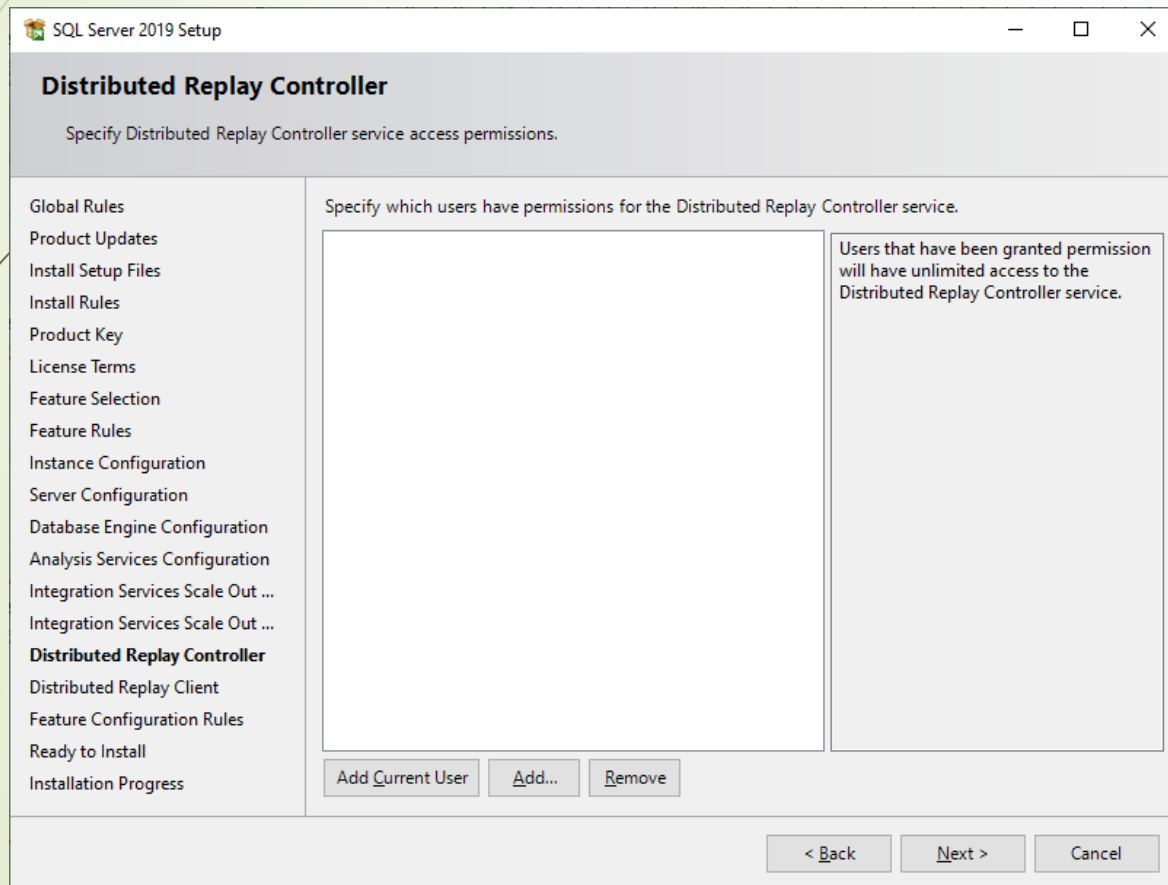
შემდეგ ფანჯარაზე გადასასვლელად უნდა გააქტიურდეს ღილაკი Next.



# MS SQL Server-ის ინსტალირება Windows-ში

29

შემდგომ დიალოგურ ფანჯარაში მომხმარებელმა აუცილებლად უნდა მიუთითოს MS SQL Server-ში განაწილების კონტროლერის სერვისების მთავარი მომხმარებლები (მაგალითად, ღილაკის Add Current User გააქტიურებით დაემატება მიმდინარე მომხმარებელი):



შემდეგ ფანჯარაზე  
გადასასვლელად უნდა  
გააქტიურდეს ღილაკი Next.

# MS SQL Server-ის ინსტალირება Windows-ში

შემდგომ დიალოგურ ფანჯარაში მომხმარებელს ეძლევა შესაძლებლობა წინა ფანჯარაში მითითებულ MS SQL Server-ში განსაზღვროს განაწილების კონტროლერის მანქანის სახელი, სამუშაო და შედეგების საქაღალდეები:

SQL Server 2019 Setup

## Distributed Replay Client

Specify the corresponding controller and data directories for the Distributed Replay Client.

Global Rules  
Product Updates  
Install Setup Files  
Install Rules  
Product Key  
License Terms  
Feature Selection  
Feature Rules  
Instance Configuration  
Server Configuration  
Database Engine Configuration  
Analysis Services Configuration  
Integration Services Scale Out ...  
Integration Services Scale Out ...  
Distributed Replay Controller  
**Distributed Replay Client**  
Feature Configuration Rules  
Ready to Install  
Installation Progress

Specify controller machine name and directory locations.

Controller Name:

Working Directory:  ...

Result Directory:  ...

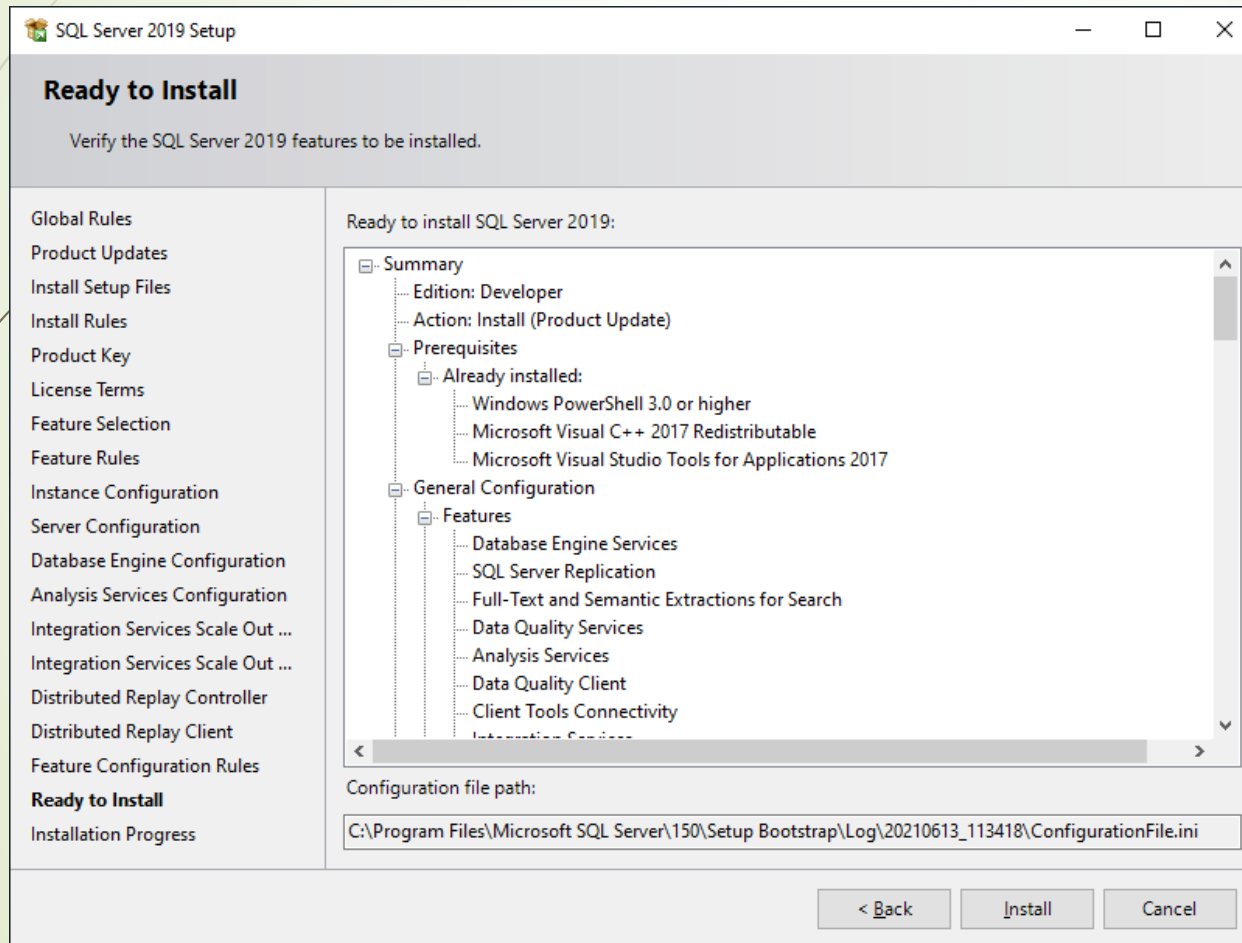
< Back   Next >   Cancel

შემდეგ ფანჯარაზე გადასასვლელად უნდა გააქტიურდეს ღილაკი Next.

# MS SQL Server-ის ინსტალირება Windows-ში

31

შემდგომ დიალოგურ ფანჯარაში მომხმარებელს ეძლევა შესაძლებლობა დაათვალიეროს MS SQL Server-ის ინსტალირების გეგმა და თანხმობის შემთხვევაში გაააქტიუროს ღილაკი Install ინსტალირების დასაწყებად:



# SQL Server Management Studio-ს ინსტალირება

რაც შეეხება SQL Server Management Studio (SSMS)-ს, რომელიც არის MS SQL Server-ის ადმინისტრირებისა და მართვის ინტეგრირებული გარემო, იგი უნდა იქნეს გადმოწერილი შემდეგი მისამართიდან:

<https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver15>

სხვა ენაზე მოქმედი SSMS შესაძლებელია გადმოწერილ იქნეს იმავე ვებ-გვერდის ბოლოში განთავსებულ ნაწილში Available languages:

## Available languages

This release of SSMS can be installed in the following languages:

SQL Server Management Studio 18.9.1:

[Chinese \(Simplified\)](#) | [Chinese \(Traditional\)](#) | [English \(United States\)](#) | [French](#) | [German](#) | [Italian](#) | [Japanese](#) | [Korean](#) | [Portuguese \(Brazil\)](#) | [Russian](#) | [Spanish](#)

ხოლო წინა ვერსიების გადმოსაწერად გამოყენებულ უნდა იქნეს უფრო ქვემოთ განლაგებული Previous versions:

## Previous versions

This article is for the latest version of SSMS only. To download previous versions of SSMS, visit [Previous SSMS releases](#).

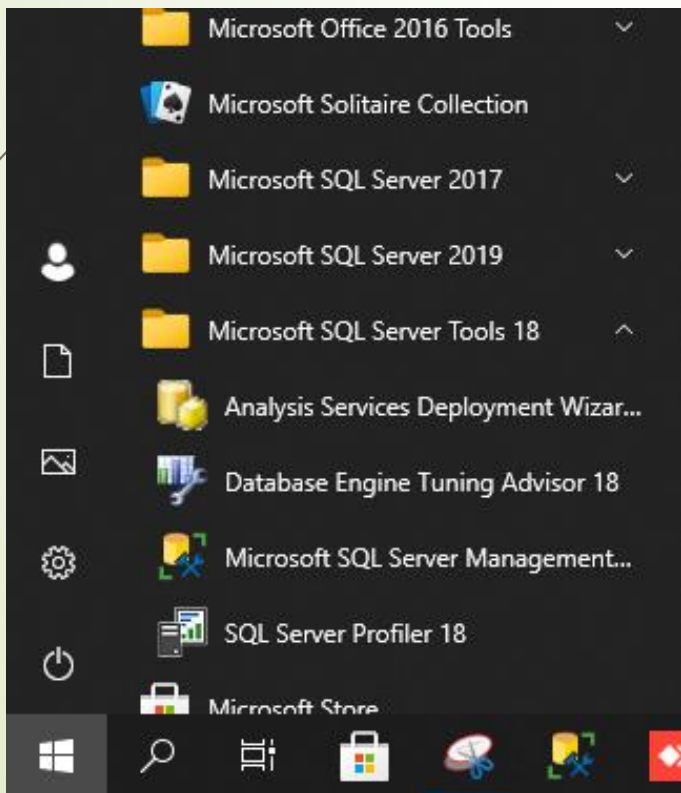


# SQL Server Management Studio-ს ინსტალირება

33

სტრიქონის გააქტიურების შედეგად ჩამოიტვირთება ფაილი „SSMS-Setup-ENU.exe“, რომელიც უნდა გაეშვას სისტემაში ადმინისტრატორის უფლებებით და ინსტალირებულ იქნეს სისტემაში სტანდარტული (ძირითადად Next) დილაგების გამოყენებით.

ინსტალირების დასრულების შემდგომ ოპერაციული გარემო მოითხოვს კომპიუტერის გადატვირთვას, რასაც მომხმარებელი უნდა დაეთანხმოს.



სისტემის ახლად ჩატვირთვის შემდგომ MS Windows 10-ის სტარტ-მენიუში მომხმარებელმა უნდა მოიძიოს ჯგუფი Microsoft SQL Server Tools 18, ჩამოშალოს იგი და გაააქტიუროს სტრიქონი Microsoft SQL Server Management Studio 18.

ამ სტრიქონის გააქტიურებისას გაეშვება ზემოაღნიშნული პროგრამა.

# 2 LEQC

# SQL Server Management Studio-ში ავთენტიფიკაცია

SQL Server Management Studio (SSMS) არის SQL Server-ის მართვისა და ადმინისტრირების გარემო. Microsoft SQL Server Management Studio 18-ს გააქტიურებისას იტვირთება შესაბამისი გარემო და იხსნება დიალოგური ფანჯარა:

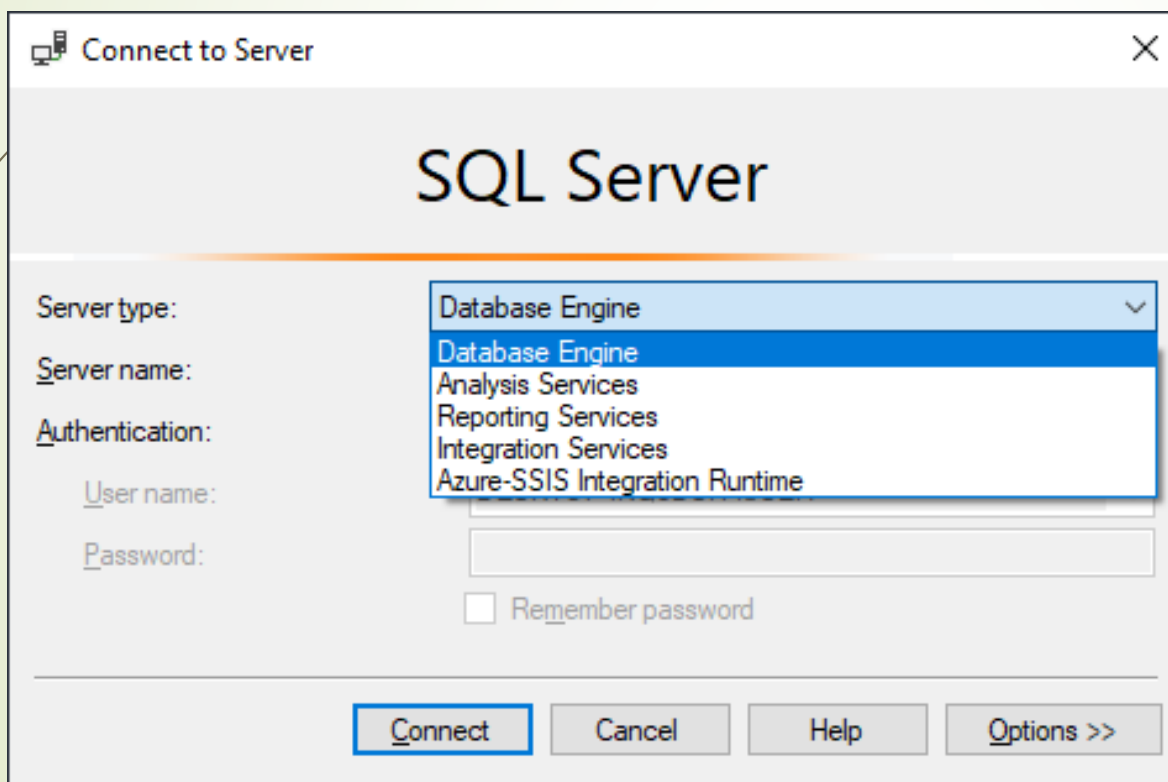
The screenshot shows the 'Connect to Server' dialog box with the following fields and values:

- Server type: Database Engine
- Server name: localhost\NA
- Authentication: Windows Authentication
- User name: DESKTOP-INQ5DSH\USER
- Password: (empty)
- ☐ Remember password

The 'Connect' button is highlighted with a blue border.

# SQL Server Management Studio-ში ავთენტიფიკაცია

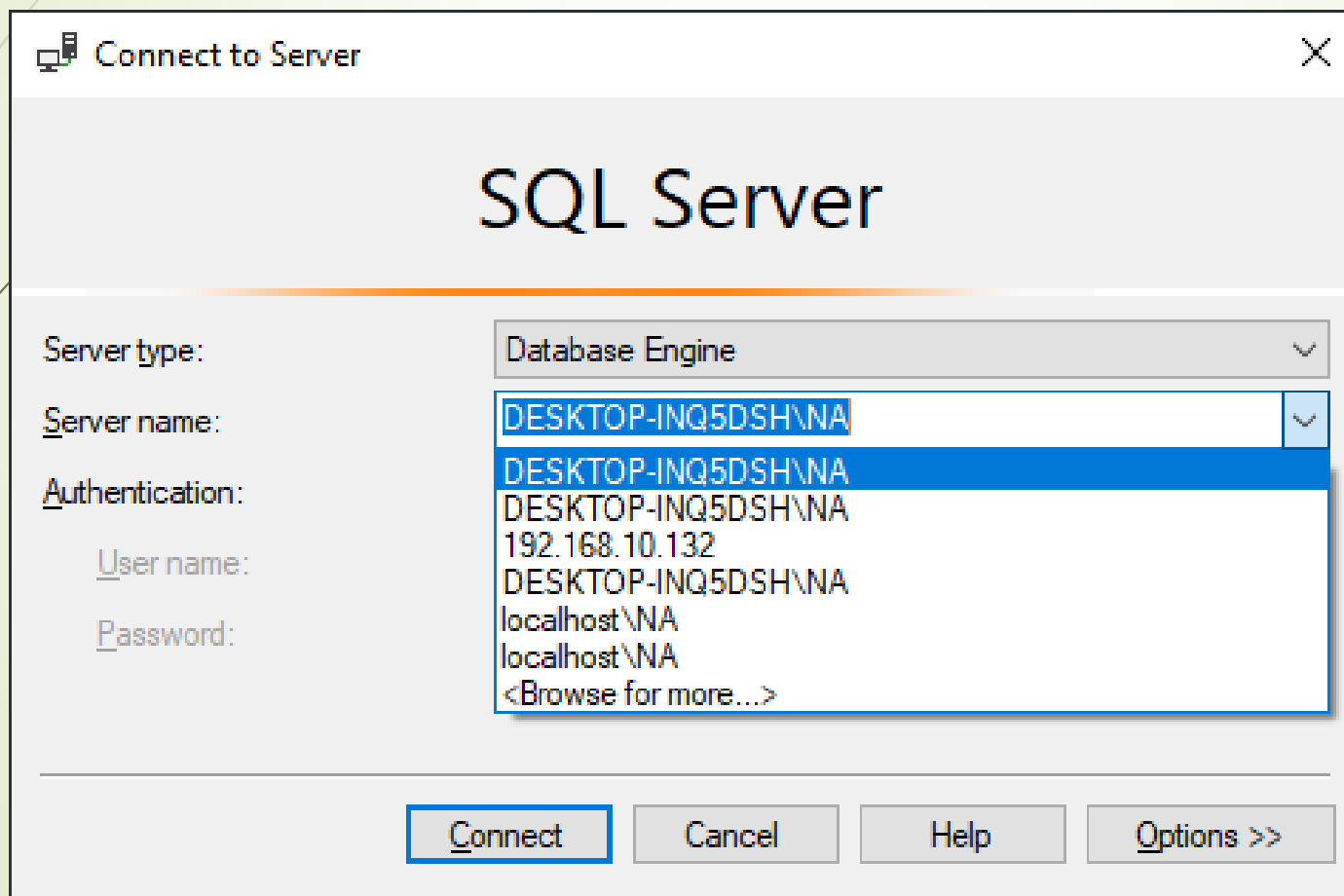
SQL Server Management Studio (SSMS)-ს გარემოს დიალოგური ფანჯრის პირველ სტრიქონში უნდა მიეთითოს თუ SQL Server-ის რომელ ტიპთან სურს მომხმარებელს მუშაობა: თვით მონაცემთა ბაზის ძირითად ნაწილთან, ანალიზების სერვისებთან, ანგარიშების სერვისებთან, ინტეგრირებულ სერვისებთან თუ Azure\_SSIS-ში ინტეგრაციის შესრულების გარემოსთან (უნდა მიეთითოს მონაცემთა ბაზის ძირითად ნაწილთან):





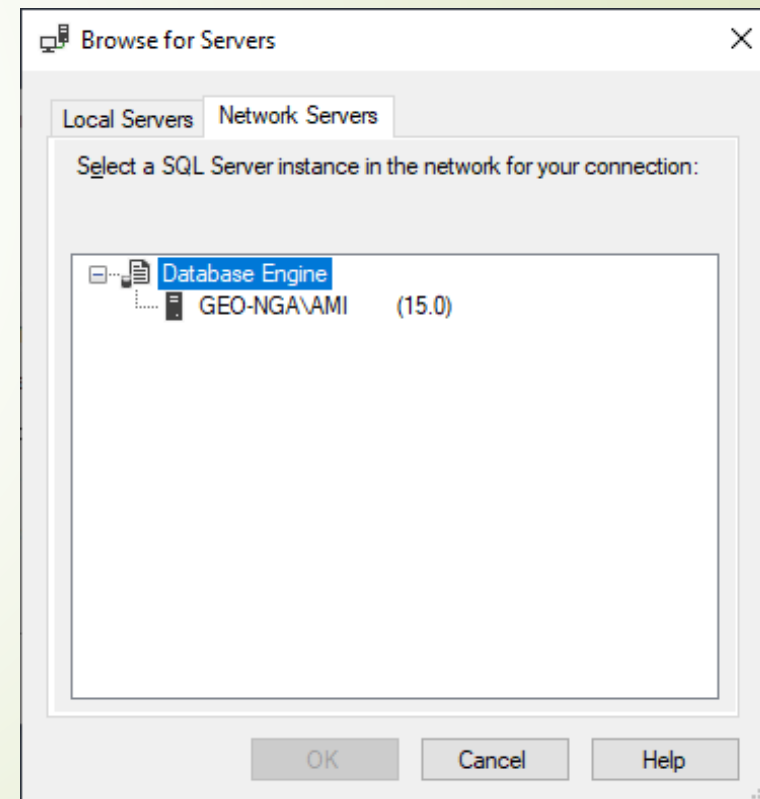
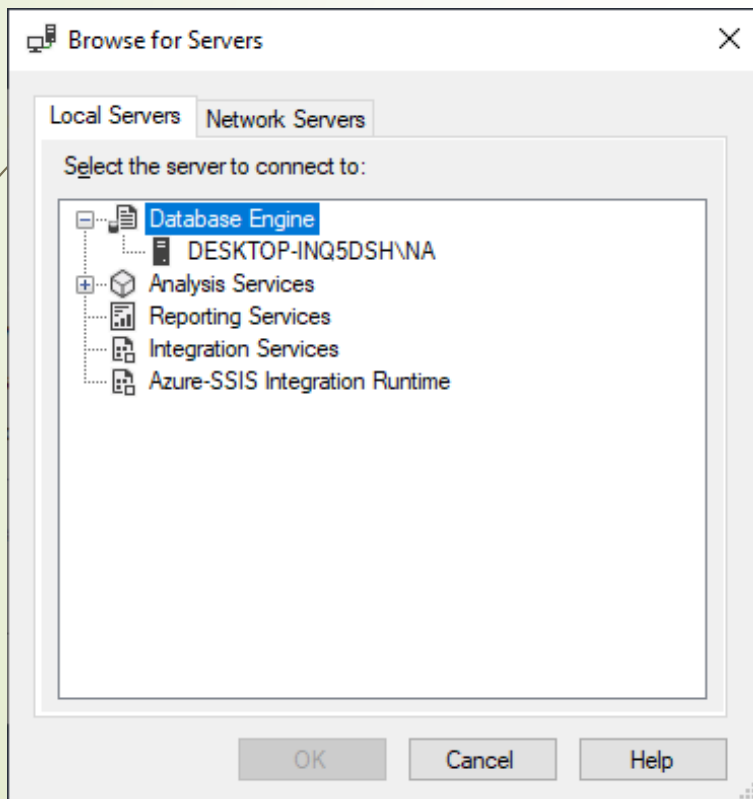
# SQL Server Management Studio-ში ავთენტიფიკაცია

SQL Server Management Studio (SSMS)-ს გარემოს დიალოგური ფანჯრის მეორე სტრიქონში უნდა მიეთითოს SQL Server-ის რომელ მონაცემთა ბაზასთან სურს მომხმარებელს მუშაობა (უნდა მიეთითოს მონაცემთა ბაზების სიიდან ერთერთი):



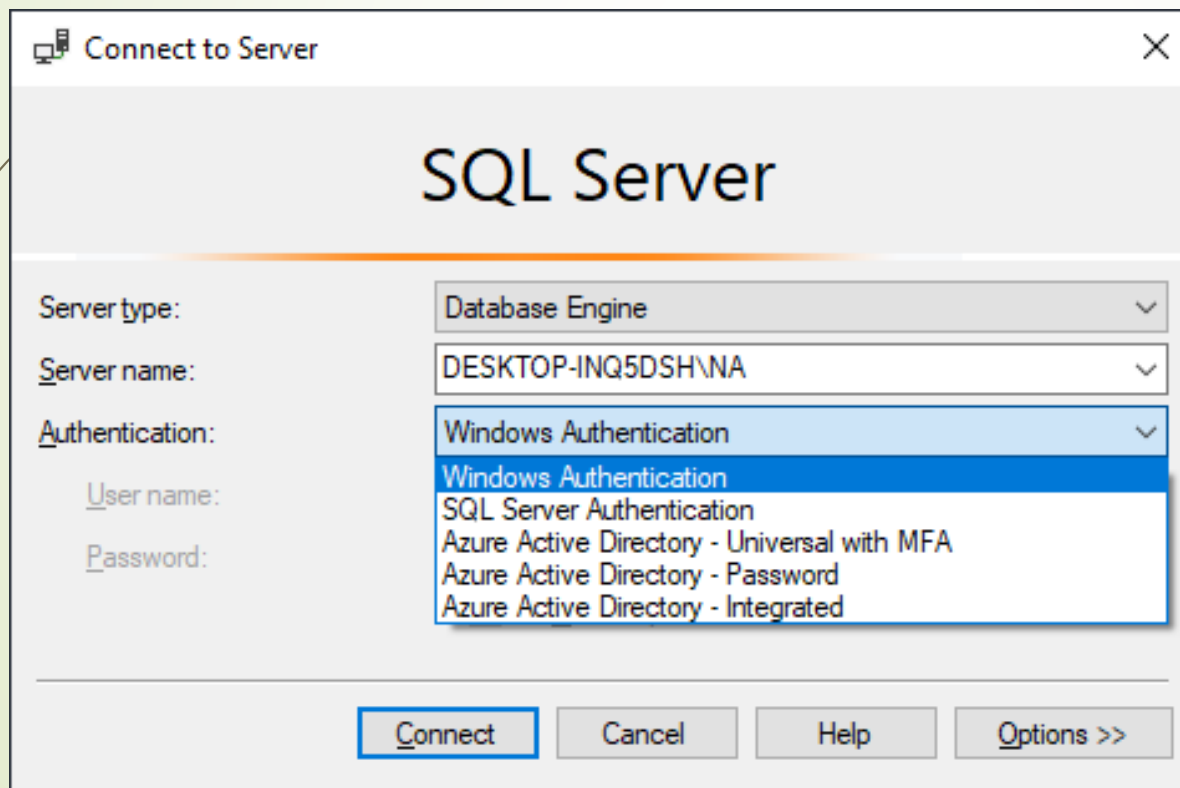
# SQL Server Management Studio-ში ავთენტიფიკაცია

თუ მონაცემთა ბაზების სიაში არ არის სასურველი, მაშინ მომხმარებელმა უნდა გაააქტიუროს სტრიქონი „Browse for more...“ და ახლად გამოსახულ ფანჯარაში შეარჩიოს „Local Servers“ ლოკალური (არსებულ კომპიუტერზე დაინსტალირებული) ან ქსელური „Network Servers“ (კომპიუტერულ ქსელში მდებარე სხვა კომპიუტერზე დაინსტალირებული) MS SQL Server-ის მონაცემთა ბაზა:



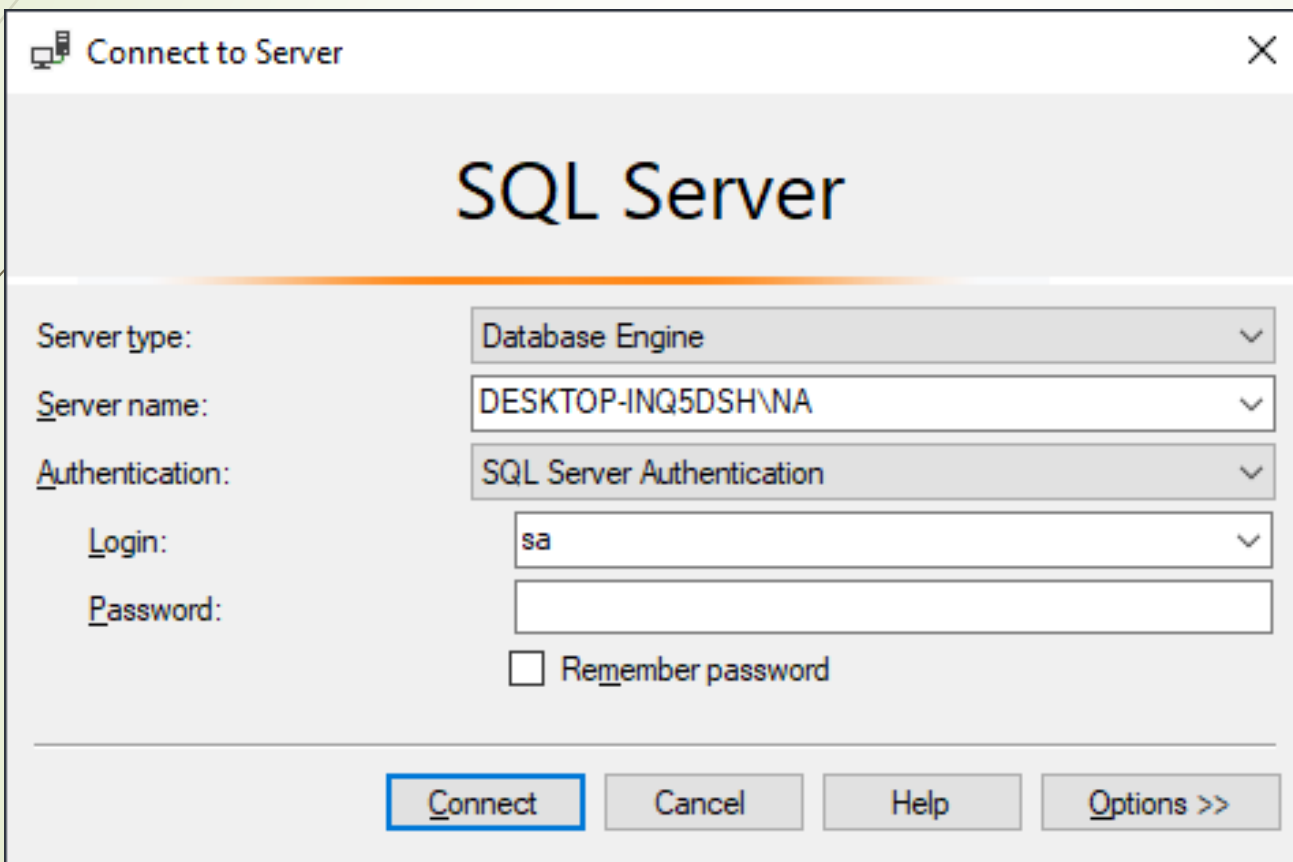
# SQL Server Management Studio-ში ავთენტიფიკაცია

SQL Server Management Studio (SSMS)-ს გარემოს დიალოგური ფანჯრის მესამე სტრიქონში უნდა მიეთითოს SQL Server-ში წვდომის ავტორიზაციის ტიპი: Windows ოპერაციულ გარემოში არსებული მომხმარებლით ავტორიზაცია „Windows Authentication“, MS SQL Server-ში არსებული მომხმარებლით ავტორიზაცია „SQL Server Authentication“ ან Azure-ში არსებული მომხმარებლით ავტორიზაცია „Azure Active Directory - ...“:



# SQL Server Management Studio-ში ავთენტიფიკაცია

SQL Server Management Studio (SSMS)-ს გარემოს დიალოგური ფანჯრის მეოთხე და მეხუთე სტრიქონები საჭიროებისამებრ გამოიყენება SQL Server-ში წვდომის ავტორიზაციისათვის მომხმარებლის სახელისა „User name“ და პაროლის „Password“ მისათითებლად:



Connect to Server

## SQL Server

Server type: Database Engine

Server name: DESKTOP-INQ5DSH\NA

Authentication: SQL Server Authentication

Login: sa

Password:

☐ Remember password

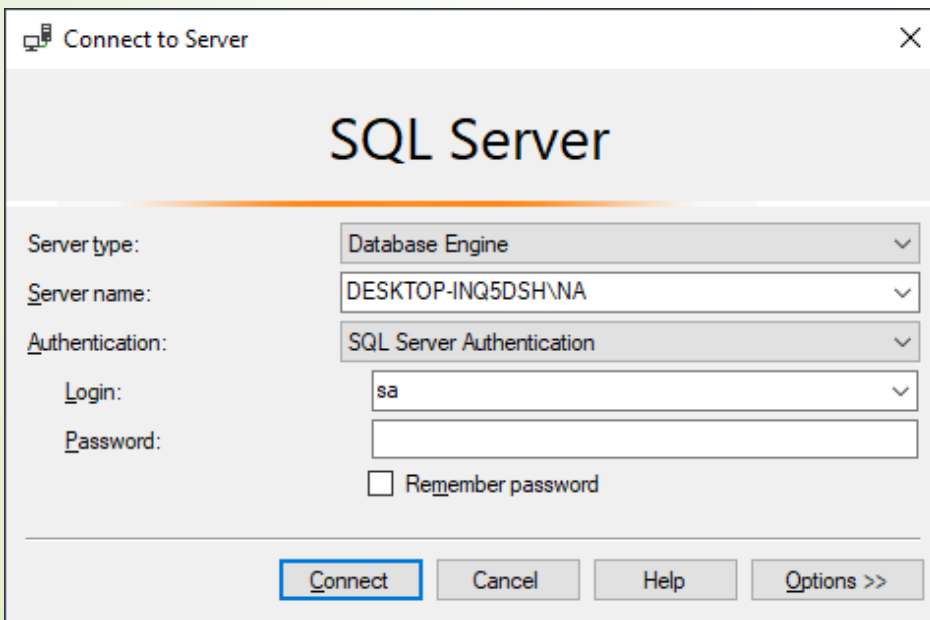
Connect Cancel Help Options >>



# SQL Server Management Studio-ში ავთენტიფიკაცია

SQL Server Management Studio (SSMS)-ს გარემოს დიალოგური ფანჯრის ღილაკების დანიშნულება შემდეგია:

- „Connect“ - შერჩეულ მონაცემთა ბაზასთან მიერთება;
- „Cancel“ - მიერთების უარყოფა;
- „Help“ - დახმარების გამოძახება;
- „Options“ - შერჩეულ მონაცემთა ბაზასთან მიერთების დამატებითი ოფციები.



Connect to Server

SQL Server

Server type: Database Engine

Server name: DESKTOP-INQ5DSH\NA

Authentication: SQL Server Authentication

Login: sa

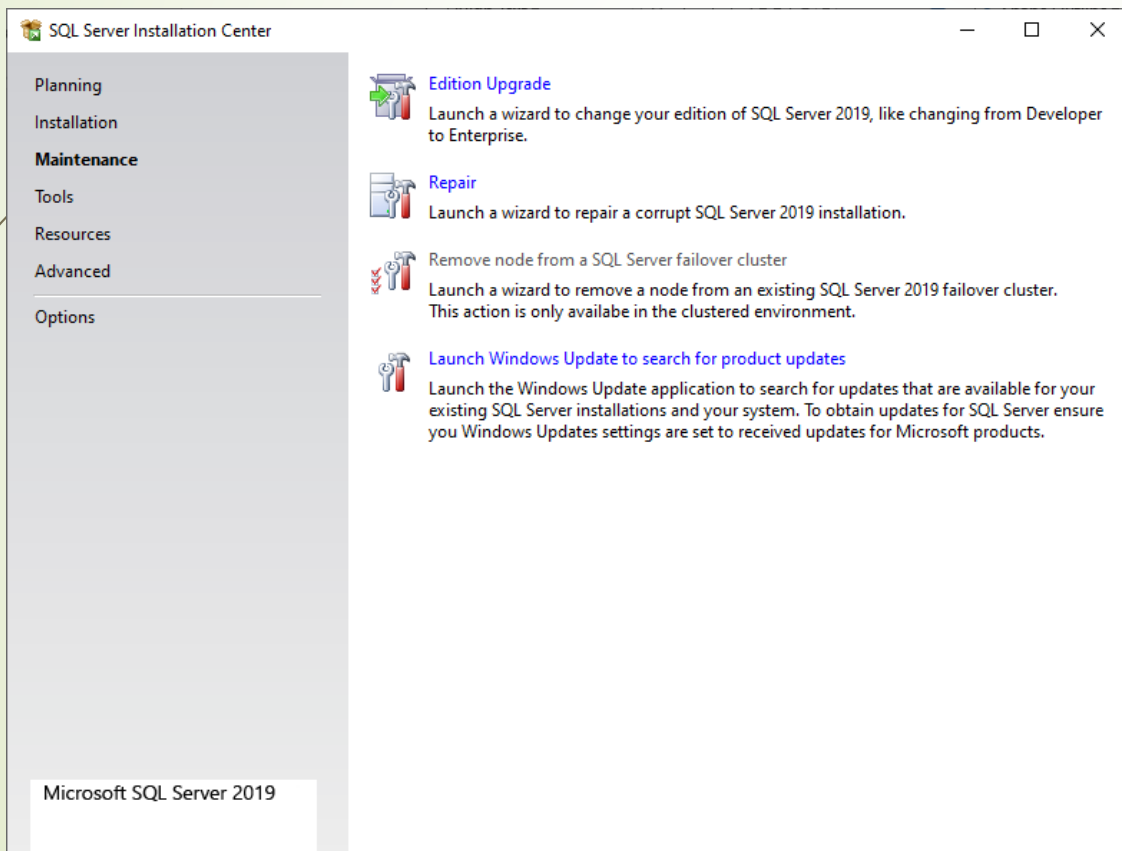
Password:

☐ Remember password

Connect Cancel Help Options >>

# MS SQL Server-ის ვერსიის განახლება ან აღდგენა

MS SQL Server-ის ინსტალირებისას setup.exe-ს გაშვებისას გამოჩენილ დიალოგურ ფანჯრის მარცხენა მიდამოში მდებარე სიაში უნდა გააქტიურდეს მესამე სტრიქონი Maintenance, რის შემდეგაც მარჯვენა მიდამოში პირველი სტრიქონის Edition Upgrade-ს გააქტიურებით განხორციელდება ვერსიის განახლება.



ამავე დიალოგური ფანჯრის მარჯვენა მიდამოს მეორე სტრიქონის გააქტიურებით ხორციელდება დაზიანებული SQL Server-ის აღდგენა.

რამდენიმე დიალოგურ ფანჯარასთან დათანხმების შემდეგ ეკრანზე გამოისახება ოპერაციის წარმატებით დასრულების გვერდი.

აღდგენა, ასევე, შესაძლებელია PowerShell-ში შემდეგი ბრძანებით:

```
Setup.exe /q /ACTION=Repair  
/INSTANCENAME=
```

**<ეგზემპლიარის\_სახელი>**

# MS SQL Server-ის კომპიუტერის გადარქმევა

43

თუ კომპიუტერს, რომელზეც არის ინსტალირებული SQL Server-ი, გადაერქვა სახელი, უნდა შესრულდეს შემდეგი პროცედურები:

```
EXEC sp_dropserver '<ძველი_სახელი>';  
GO  
EXEC sp_addserver '<ახალი_სახელი>', local;  
GO
```

რის შემდეგაც SQL Server-ი უნდა გადაიტვირთოს.

თუ კომპიუტერს, რომელზეც არის ინსტალირებული SQL Server-ის სახელობითი ეგზემპლარი, გადაერქვა სახელი, უნდა შესრულდეს შემდეგი პროცედურები:

```
EXEC sp_dropserver '<ძველი_სახელი\ეგზემპლარის_სახელი>';  
GO  
EXEC sp_addserver '<ახალი_სახელი\ეგზემპლარის_სახელი>', local;  
GO
```

რის შემდეგაც SQL Server-ი უნდა გადაიტვირთოს.

შემდეგი ფუნქცია ამოწმებს სერვერის სახელს

```
SELECT @@SERVERNAME AS 'სერვერის_სახელი'
```

# MS SQL Server-ის ინსტალირება SysPrep-ით

## MS SQL Server-ის ინსტალირება SysPrep-ის გამოყენებით.

MS SQL Server-ი შესაძლებელია დაინსტალირდეს ასევე SysPrep-ის გამოყენებით.

SysPrep-ი იძლევა SQL Server-ის ცალკეული ეგზემპლიარის მომზადების შესაძლებლობას, რომლის გამოყენებით მოგვიანებით სხვადასხვა კომპიუტერზე შესაძლებელია კონფიგურირების დასრულება. ეს მეთოდი მოიცავს ორეტაპიან პროცესს:

- **იმიჯის მომზადება** - ამ ეტაპზე ორობითი ფაილების დაყენების შემდგომ, კომპიუტერის, ქსელის ან ინფორმაციის კონფიგურირების გარეშე ინსტალირების პროცესი ჩერდება, რომელიც მიბმულია მხოლოდ SQL Server-ის მოსამზადებელ ეგზემპლიარზე;
- **იმიჯის შექმნის დასრულება** - ამ ეტაპზე სრულდება SQL Server-ის მომზადებული ეგზემპლიარის კონფიგურირება, რომლის დროსაც ეთითება ინფორმაცია კონკრეტული კომპიუტერის, ქსელის ან საადრიცხვო ჩანაწერის.

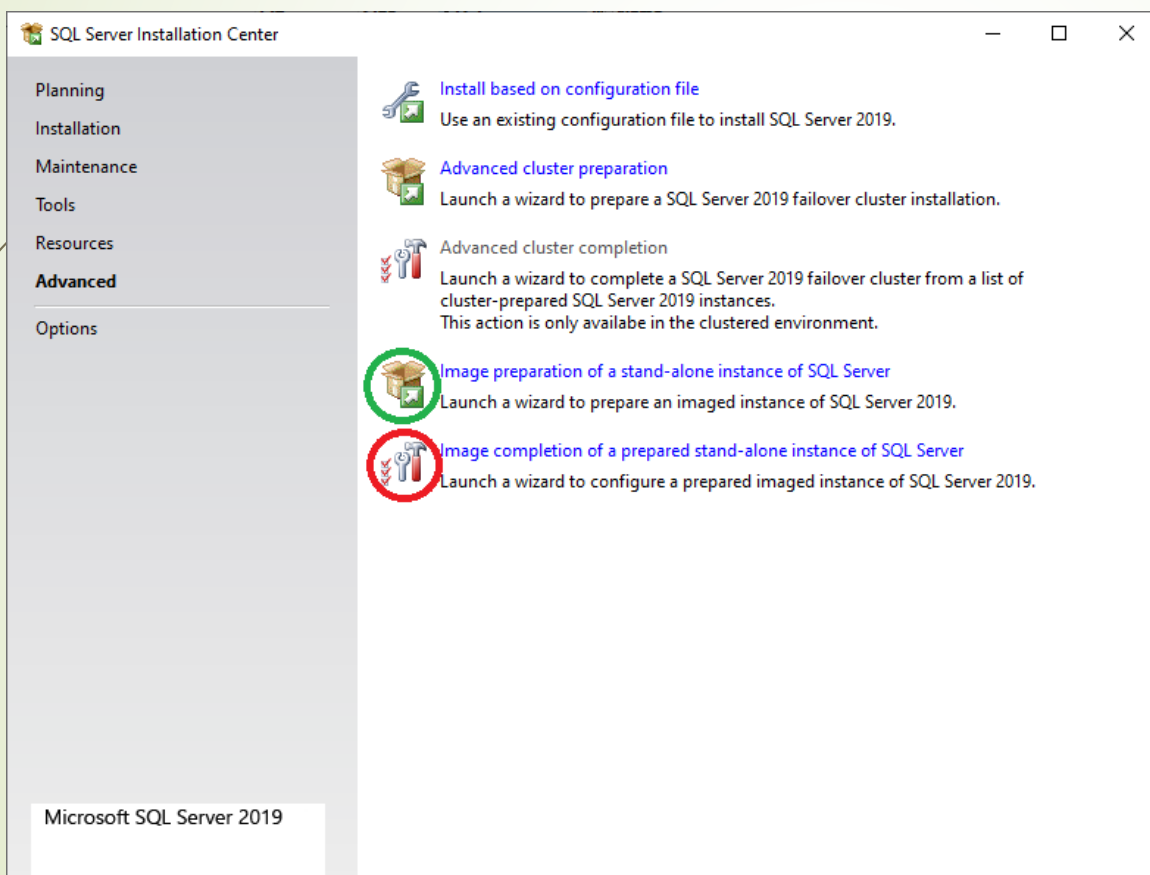
ძირითადად SQL Server SysPrep-ი გამოიყენება შემდეგ შემთხვევებში:

- წინასწარ იქნეს მომზადებული ერთი ან რამდენიმე არაკონფიგურირებული იმიჯი, რომლის კონფიგურირების დასრულება განხორციელდება სხვადასხვა კომპიუტერზე;
- Windows SysPrep-ით Windows-ის იმიჯის მომზადებისას მიუერთდეს SQL Server SysPrep-იც და კომპიუტერებზე დაინსტალირდეს მათი კომბინაცია ერთად.



# MS SQL Server-ის ინსტალირება SysPrep-ით

SQL Server SysPrep-ის იმიჯის შესაქმნელად MS SQL Server-ის ინსტალირებისას setup.exe-ს გაშვებისას გამოჩენილ დიალოგურ ფანჯრის მარცხენა მიდამოში მდებარე სიაში უნდა გააქტიურდეს ბოლოდან მეორე სტრიქონი Advanced, რის შემდეგაც მარჯვენა მიდამოში უნდა გააქტიურდეს ბოლოდან მეორე სტრიქონი Image preparation of a



stand-alone instance of SQL Server (მწვანე წრეში), ხოლო SQL Server SysPrep-ის შექმნილი იმიჯის კონფიგურირების დასრულებისათვის და MS SQL Server-ის ინსტალირებისათვის იგივე დიალოგურ ფანჯრის მარჯვენა მიდამოში უნდა გააქტიურდეს ბოლო სტრიქონი Image completion of a prepared stand-alone instance of SQL Server (წითელ წრეში). სტრიქონების გააქტიურების შემდგომი მოქმედებები უკვე აღწერილია წინა სლაიდებში, გარდა იმ მოქმედებებისა, რომლებიც აღწერილია შემდეგ სლაიდებში.

# MS SQL Server-ის ინსტალირება SysPrep-ით

46

## SQL Server SysPrep-ის იმიჯის შექმნის დიალოგური ფანჯრის განსხვავებული გვერდები:

- SQL Server SysPrep-ის იმიჯის შექმნისას დიალოგური ფანჯრის Prepare Image Type გვერდში უნდა მოინიშნოს Prepare a new instance of SQL Server. გასათვალისწინებელია, რომ აღნიშნული გვერდი იქნება ნაჩვენები მხოლოდ მაშინ, როდესაც მომზადება გამოსახულება ტიპი გვერდზე არის ნაჩვენები მხოლოდ მაშინ, როდესაც მიმდინარე კომპიუტერში არსებობს არაკონფიგურირებული მომზადებული SQL Server ეგზემპლარი. შესაძლებელია მომზადებულ იქნეს SQL Server-ის ახალი ეგზემპლარი ან არსებულ ეგზემპლარს დაემატოს SysPrep-ის მხარდაჭერილი კომპონენტები;
- SQL Server SysPrep-ის იმიჯის შექმნისას დიალოგურ ფანჯრის Prepare Image Rules გვერდზე კონფიგურაცია ამოწმებს კომპიუტერის სისტემის მდგომარეობას, რის შემდეგაც გრძელდება იმიჯის შექმნა;
- SQL Server SysPrep-ის იმიჯის შექმნისას დიალოგურ ფანჯრის Instance Configuration გვერდზე უნდა იქნეს მითითებული ეგზემპლარის ID. აქვე შესაძლებელია მითითებულ იქნეს იმიჯის ძირითადი საქაღალდე (Instance root directory).

# MS SQL Server-ის ინსტალირება SysPrep-ით

47

SQL Server SysPrep-ის შექმნილი იმიჯის კონფიგურირების დასრულების დიალოგური ფანჯრის განსხვავებული გვერდები:

- SQL Server SysPrep-ის შექმნილი იმიჯის კონფიგურირების დასრულების დიალოგური ფანჯრის „მხარდამჭერი ფაილების დაყენების“ (Setup Support Files) გვერდზე უნდა გააქტიურდეს ღილაკი Install ამ ფაილების ინსტალირებისათვის;
- SQL Server SysPrep-ის შექმნილი იმიჯის კონფიგურირების დასრულების დიალოგური ფანჯრის „მომზადებული ეგზემპლიარის შერჩევა“ (Select a Prepared Instance) ჩამოსაშლელ სიიდან უნდა იქნეს შერჩეული ის ეგზემპლიარი, რომლის დასრულებაც არის გათვალისწინებული.



# MS SQL Server-ის ინსტალირება SysPrep-ით

## SQL Server-ის გამზადებული ეგზემპლარისათვის კომპონენტების დამატება

SQL Server-ის გამზადებული ეგზემპლარისათვის კომპონენტების დასამატებლად უნდა განხორციელდეს შემდეგი განსხვავებული მოქმედებები:

- MS SQL Server-ის ინსტალირებისას setup.exe-ს გაშვებისას გამოჩენილ დიალოგურ ფანჯრის მარცხენა მიდამოში მდებარე სიაში უნდა გააქტიურდეს ბოლოდან მეორე სტრიქონი Advanced, რის შემდეგაც მარჯვენა მიდამოში უნდა გააქტიურდეს ბოლოდან მეორე სტრიქონი Image preparation of a stand-alone instance;
- SQL Server-ის გამზადებული ეგზემპლარისათვის კომპონენტების დამატების დიალოგური ფანჯრის Prepare Image Type გვერდში უნდა მოინიშნოს Add features to an existing prepared instance of SQL Server. ჩამოსაშლელ სიიდან უნდა იქნეს შერჩეული ის ეგზემპლარი, რომელშიც კომპონენტები დამატება არის გათვალისწინებული;
- SQL Server-ის გამზადებული ეგზემპლარისათვის კომპონენტების დამატების დიალოგური ფანჯრის Feature Selection გვერდის გამოყენებით უნდა მოინიშნოს დასამატებელი კომპონენტები.



# MS SQL Server-ის ინსტალირება SysPrep-ით

## SQL Server-ის გამზადებული ეგზემპლარიდან კომპონენტების წაშლა

SQL Server-ის გამზადებული ეგზემპლარისათვის კომპონენტების წასაშლელად უნდა განხორციელდეს შემდეგი განსხვავებული მოქმედებები:

- Windows-ის ოპერაციული გარემოს მართვის პანელში (Control Panel) უნდა გააქტიურდეს „პროგრამები და კომპონენტები“ (Program and Features), რომელშიც უნდა მიეთითოს SQL Server-ის სისტემიდან „წაშლა“ (Remove);
- გამოჩენილ დიალოგურ ფანჯრის „ეგზემპლარის შერჩევის“ (Select Instance) გვერდზე აირჩიეთ SQL Server-ის მომზადებული ეგზემპლარი, რომლიდანაც უნდა განხორციელდეს კომპონენტების წაშლა;
- გამოჩენილ დიალოგურ ფანჯრის „კომპონენტების შერჩევის“ (Select Features) გვერდზე აირჩიეთ კომპონენტები, რომლებიც უნდა წაიშალოს SQL Server-ის მომზადებულ ეგზემპლარიდან.

# MS SQL Server-ის ინსტალირება SysPrep-ით

## SQL Server-ის მტყუნებამედები კლასტერის მომზადება (დამოუკიდებლად)

SQL Server SysPrep-ის იმიჯის შექმნა Windows SysPrep-ის Windows-ის იმიჯში კომბინაციით, შესაძლებელია ასევე Windows PowerShell-ის ბრძანებითი სტრიქონის გამოყენებით, ამისათვის საჭიროა შემდეგი მოქმედებები:

- უნდა შეიქნას SQL Server SysPrep-ის იმიჯი შემდეგი მაგალითის შესაბამისად:  
**Setup.exe /q /ACTION=PrepareImage I /FEATURES=SQLEngine /InstanceID=<MYINST> /IACCEPTSQLSERVERLICENSETERMS**
- უნდა განთავსდეს SQL Server SysPrep-ის იმიჯი Windows SysPrep Specialize-ს გამოყენებით;
- უნდა შევიდეთ Windows Failover Cluster-ში;
- ყველა კვანძისათვის უნდა გაეშვას MS SQL Server-ის საინსტალაციო ფაილი setup.exe პარამეტრით /ACTION = PrepareFailoverCluster შემდეგი მაგალითის შესაბამისად:  
**setup.exe /q /ACTION=PrepareFailoverCluster  
 /InstanceName=<ეგზემპლარის\_სახელი> /Features=SQLEngine  
 /SQLSVCAccount="<დომეინის\_სახელი\მომზადებლის\_სახელი>"  
 /SQLSVCPASSWORD="\*\*\*\*\*" /IACCEPTSQLSERVERLICENSETERMS**

# MS SQL Server-ის ინსტალირება SysPrep-ით

51

SQL Server-ის მტყუნებამედები კლასტერის შექმნა (ავტომატური ინსტალირებით)

უნდა გაეშვას MS SQL Server-ის საინსტალაციო ფაილი setup.exe პარამეტრით /ACTION = CompleteFailoverCluster კვანძზე, რომელიც ეკუთვნის მოწყვლად საცავ ჯგუფს შემდეგი მაგალითის შესაბამისად:

**setup.exe /q /ACTION=CompleteFailoverCluster**

**/InstanceName=<ეგზემპლარის\_სახელი> /FAILOVERCLUSTERDISKS="<დისკის\_სახელი  
მაგალითად, 'Disk S:'>:" /FAILOVERCLUSTERNETWORKNAME="<ქსელის\_სახელი>"**

**/FAILOVERCLUSTERIPADDRESSES="IPv4;xx.xxx.xx.xx;Cluster Network;xxx.xxx.xxx.x"**

**/FAILOVERCLUSTERGROUP="ჯგუფის\_სახელი"**

**/INSTALLSQLDATADIR="<Drive>:\<Path>\საქაღალდის\_სახელი"**

**/SQLCOLLATION="ენის\_კოდირება მაგალითად, SQL\_Latin1\_General\_CP1\_CS\_AS"**

**/SQLSYSADMINACCOUNTS="<დომეინის\_სახელი\მომხმარებლის\_სახელი>"**

# MS SQL Server-ის ინსტალირება SysPrep-ით

SQL Server-ის მტყუნებამდე კლასტერს კვანძის დამატება (ავტომატური ინსტალირებით)

- უნდა განთავსდეს SQL Server SysPrep-ის იმიჯი Windows SysPrep Specialize-ს გამოყენებით;
- უნდა შევიდეთ Windows Failover Cluster-ში;
- ყველა კვანძისათვის უნდა გაეშვას MS SQL Server-ის საინსტალაციო ფაილი setup.exe პარამეტრით /ACTION = AddNode შემდეგი მაგალითის შესაბამისად:  
**setup.exe /q /ACTION=AddNode /InstanceName=<ეგზემპლარის\_სახელი>  
/Features=SQLEngine  
/SQLSVCAccount="<დომეინის\_სახელი\მომხმარებლის\_სახელი>"  
/SQLSVCPASSWORD="\*\*\*\*\*" /IACCEPTSQLSERVERLICENSETERMS**



# MS SQL Server-ის ინსტალირება PowerShell DSC-ით

Desired State Configuration-ის (DSC) გამოყენებით მომხმარებელს ეძლევა შესაძლებლობა შექმნას SQL Server-ის ერთი საინსტალაციო კონფიგურაციის შაბლონი და გამოიყენოს იგი ასიათასობით სერვერზე. ინსტალირებისას საჭირო იქნება კომპიუტერისაგან გამომდინარე კონფიგურაციის მხოლოდ რამდენიმე პარამეტრის შეცვლა.

მაგალითისათვის განვიხილოთ SQL Server 2017-ის დამოუკიდებელი ეგზემპლარის Windows Server 2016-ზე საწყის ინსტალირება DSC რესურსის SqlServerDsc-ის გამოყენებით, რისთვისაც საჭიროა:

- კომპიუტერი, რომელზეც ინსტალირებულია Windows Server 2016;
- SQL Server 2017 საინსტალაციო პაკეტი, რომელიც მაგალითისათვის მოვათავსოთ ლოგიკურ დისკზე:

**C:\en\_sql\_server\_2017\_enterprise\_x64\_dvd\_11293666.iso**

- DSC-ის რესურსი SqlServerDsc. SqlServerDsc-ის საინსტალაციო პაკეტი მდებარეობს ვებ-გვერდზე <https://www.powershellgallery.com/packages/SqlServerDsc/15.2.0>, რომელიც გადმოწერის შემდეგ ინსტალირდება Windows-ის ბრძანებათა სტრიქონიდან

**Install-Module -Name SqlServerDsc**

# MS SQL Server-ის ინსტალირება PowerShell DSC-ით

SQL Server 2017 საინსტალაციო პაკეტი (ISO) მოვითავსოთ საქაღალდეში:

```
New-Item -Path C:\SQL2017 -ItemType Directory  
$mountResult = Mount-DiskImage -ImagePath  
'C:\en_sql_server_2017_enterprise_x64_dvd_11293666.iso' -PassThru  
$volumeInfo = $mountResult | Get-Volume  
$driveInfo = Get-PSDrive -Name $volumeInfo.DriveLetter  
Copy-Item -Path ( Join-Path -Path $driveInfo.Root -ChildPath '*' ) -Destination  
C:\SQL2017\ -Recurse  
Dismount-DiskImage -ImagePath  
'C:\en_sql_server_2017_enterprise_x64_dvd_11293666.iso'
```

შევქმნათ კონფიგურაციის ფუნქცია, რომელიც გამოიძახება „მართული ობიექტის ფორმატის“ (Managed Object Format - MOF) დოკუმენტების შესაქმნელად :

```
Configuration SQLInstall  
{...}
```

განვახორციელოთ მოდულების იმპორტი მიმდინარე სესიაში, რომლებიც მიუთითებენ კონფიგურაციის დოკუმენტს MOF დოკუმენტების შექმნის წესს, ასევე მიუთითებენ DSC ძრავას თუ როგორ იქნეს გამოყენებული MOF დოკუმენტები სერვერზე:

```
Import-DscResource -ModuleName SqlServerDsc
```

# MS SQL Server-ის ინსტალირება PowerShell DSC-ით

ასევე საჭიროა .NET Framework-45-Core-ს ინსტალირება SQL Server-ის ინსტალირებამდე, რისთვისაც გამოიყენება WindowsFeature რესურსი:

```
WindowsFeature 'NetFramework45'
{
    Name = 'Net-Framework-45-Core'
    Ensure = 'Present'
}
```

SqlSetup რესურსი ატყობინებს DSC-ს თუ როგორ უნდა დაინსტალირდეს SQL Server-ი. საბაზისო ინსტალირებისთვის საჭიროა შემდეგი პარამეტრები:

- **InstanceName** - ეგზემპლარის სახელი (ნაგულისხმევია MSSQLSERVER);
- **Features** - ინსტალირების კომპონენტები (ჩვენ მაგალითში არის მხოლოდ SQLEngine);
- **SourcePath** - გზა SQL Server-ის საინსტალაციო პაკეტთან (ჩვენ მაგალითში არის C:\SQL2017);
- **SQLSysAdminAccounts** - sysadmin როლის წევრი მომხმარებლები ან ჯგუფები (ჩვენ მაგალითში არის ლოკალური ჯგუფი Administrators).

SqlSetup რესურსი გამოიყენება მხოლოდ SQL Server-ის ინსტალირებისათვის და არა გამოყენებული პარამეტრების შენახვისათვის.

# MS SQL Server-ის ინსტალირება PowerShell DSC-ით

კონფიგურაციის დასრულებისათვის PowerShell-ში იწერება ბრძანება:

**Configuration SQLInstall**

```
{  
  Import-DscResource -ModuleName SqlServerDsc  
  node localhost  
  {  
    WindowsFeature 'NetFramework45'  
    {  
      Name = 'NET-Framework-45-Core'  
      Ensure = 'Present'  
    }  
    SqlSetup 'InstallDefaultInstance'  
    {  
      InstanceName      = 'MSSQLSERVER'  
      Features          = 'SQLENGINE'  
      SourcePath         = 'C:\SQL2017'  
      SQLSysAdminAccounts = @('Administrators')  
      DependsOn          = '[WindowsFeature]NetFramework45'  
    }  
  }  
}
```



# MS SQL Server-ის ინსტალირება PowerShell DSC-ით

კონფიგურაციის კომპილაციისათვის PowerShell-ში იწერება კონფიგურაციის სკრიპტის წყარო:

```
.. \SQLInstallConfiguration.ps1
```

და იგივე PowerShell-ში ეშვება კონფიგურაციის ფუნქცია:

## SQLInstall

სამუშაო საქაღალდეში იქმნება SQLInstall საქაღალდე, რომელიც შეიცავს ფაილს localhost.mof. MOF ფაილში განთავსებულია კომპილირებული DSC-ს კონფიგურაცია.

SQL Server-ის ინსტალირებისათვის DSC-ს მიხედვით PowerShell-ში იწერება ბრძანება:

```
Start-DscConfiguration -Path C:\SQLInstall -Wait -Force -Verbose
```

აქ არის შემდეგი პარამეტრები:

- **Path** - გზა საქაღალდის, რომელშიც მდებარეობენ MOF საბუთები (ჩვენ მაგალითში არის C:\SQLInstall);
- **Wait** - კონფიგურაციის სამუშაოს დასრულების დალოდება;
- **Force** - არსებული DSC კონფიგურაციების გადანაწილება;
- **Verbose** - გამოსავალი მონაცემების ვიზუალიზაცია.

# MS SQL Server-ის ინსტალირება PowerShell DSC-ით

ბრძანების გაშვების შემდგომ თუ არ არის შეცდომები (წითელი ტექსტი), ეკრანზე გამოისახება Operation 'Invoke CimMethod' Complete რაც ინსტალირების წარმატებით დასრულებას ნიშნავს.

ასვე წარმატებული ინსტალირება შეიძლება გადამოწმდეს PowerShell-ში ბრძანებით:

## Test-DscConfiguration

რომელმაც უნდა დააბრუნოს პასუხი **True**.

ასევე სერვისების სიაში უკვე გამოისახება SQL Server-ის სერვისები, რომელიც შეიძლება ინახოს PowerShell-ში ბრძანებით:

## Get-Service -Name \*SQL\*

შედეგად მივიღებთ სერვისების სიას:

Running	MSSQLSERVER	SQL Server (MSSQLSERVER)
Stopped	SQLBrowser	SQL Server Browser
Running	SQLSERVERAGENT	SQL Server Agent (MSSQLSERVER)
Running	SQLTELEMETRY	SQL Server CEIP service (MSSQLSERVER)
Running	SQLWriter	SQL Server VSS Writer

# MS SQL Server-ის ინსტალირების ჟურნალი

SQL Server-ი ჟურნალის ფაილებს ნაგულისხმევად ქმნის დათარიღებულ და დროში გაწერილ საქაღალდეში:

**%programfiles %\Microsoft SQL Server\nnn\Setup Bootstrap\Log\YYYYMMDD\_hhmmss**

სადაც nnn არის რიცხვი, რომლებიც შეესაბამება ინსტალირებული SQL Server-ის ვერსიას, ხოლო YYYYMMDD\_hhmmss ჟურნალის საქაღალდის სახელის ფორმატია.

ინსტალირებისას ავტომატურ რეჟიმში ხორციელდება ჟურნალების შექმნა %temp%\sqlsetup\*.log-ში, რის შემდეგაც ამ საქაღალდეს ყველა ფაილი არქივდება შესაბამის საქაღალდის Log\*.cab ფაილში.

OSDisk (C:) > Program Files > Microsoft SQL Server > 140 > Setup Bootstrap > Log				OSDisk (C:) > Program Files > Microsoft SQL Server > 140 > Setup Bootstrap > Log > 20180326_084208			
Name	Date modified	Type	Size	Name	Date modified	Type	Size
20180326_083544	3/26/2018 4:45 PM	File folder		Datstore	3/26/2018 9:08 AM	File folder	
20180326_083658	3/26/2018 8:37 AM	File folder		resources	3/26/2018 8:42 AM	File folder	
20180326_083952	3/26/2018 8:40 AM	File folder		AdvancedAnalytics_Cpu64_1.log	3/26/2018 8:57 AM	Text Document	258 KB
20180326_084157	3/26/2018 4:45 PM	File folder		ConfigurationFile.ini	3/26/2018 8:54 AM	Configuration sett...	20 KB
<b>20180326_084208</b>	3/26/2018 9:08 AM	File folder		Detail.txt	3/26/2018 9:08 AM	TXT File	4,146 KB
20180507_123534	5/7/2018 12:56 PM	File folder		HkEngineEventFile_0_13166554069583...	3/26/2018 9:07 AM	Microsoft SQL Ser...	69 KB
20180507_123541	5/7/2018 12:36 PM	File folder		HkEngineEventFile_0_13166554094530...	3/26/2018 9:08 AM	Microsoft SQL Ser...	69 KB
Summary.txt	5/7/2018 12:36 PM	TXT File		MDS_Cpu64_1.log	3/26/2018 8:58 AM	Text Document	2,732 KB



# MS SQL Server-ის ინსტალირების ჟურნალი

SQL Server-ის ინსტალირების ჟურნალის ფაილებია:

%programfiles%\Microsoft SQL Server\nnn\Setup Bootstrap\Log საქალაქდებში:

➤ **Summary.txt**

%programfiles%\Microsoft SQL Server\nnn\Setup Bootstrap\Log\YYYYMMDD\_hhmmss საქალაქდებში:

➤ **Summary\_<MachineName>\_YYYYMMDD\_hhmmss.txt**

➤ **Detail.txt**

➤ **Datastore**

➤ **MSI Log Files** (ფაილი <Name>.log)

➤ **ConfigurationFile.ini**

➤ **SystemConfigurationCheck\_Report.htm**

%temp%\ საქალაქდის sqlsetup\*.log ფაილი

➤ **For unattended installations**

განვიხილოთ ამ ფაილების დანიშნულებები:



# MS SQL Server-ის ინსტალირების ჟურნალი

## Summary.txt

ეს ფაილი აჩვენებს SQL Server-ის კომპონენტებს, რომლებიც გამოვლინდა ინსტალირების პროცესში, ოპერაციული სისტემის გარემო, ბრძანებათა სტრიქონის პარამეტრების მნიშვნელობები და თითოეული შესრულებული MSI/MSP-ს საერთო სტატუსი. ეს ყველაფერი დაყოფილია შემდეგ ნაწილებად:

- შესრულების საერთო ანგარიში;
- კომპიუტერის თვისებები და კონფიგურაცია, რომელზეც ინსტალირდება SQL Server-ი;
- კომპიუტერზე ადრე დაინსტალირებული SQL Server-ის მახასიათებლები;
- ინსტალირების პაკეტის თვისებებისა და ვერსიის აღწერა;
- ინსტალირების მითითებული შესრულების გარემოს შემავალი პარამეტრები;
- კონფიგურირების ფაილის ადგილმდებარეობა;
- მონაცემები შესრულების შედეგებზე;
- გლობალური წესები;
- ინსტალირების სცენარის სპეციფიკური წესები;
- წესები, რომლის შესრულებისას წარმოიქმნა შეცდომა;
- წესების ანგარიშის ფაილის მდებარეობა.

# MS SQL Server-ის ინსტალირების ჟურნალი

## Summary\_<MachineName>\_YYYYMMDD\_hhmmss.txt

Summary.txt ძირითადი ფაილის მსგავსია და გენერირდება ძირითადი სამუშაო პროცესის შესრულებისას.

## Detail.txt

Detail.txt გენერირდება ძირითადი მუშა პროცესის შესრულებისას, როგორცაა ინსტალირება ან განახლება და უზრუნველყოფს შესრულების დეტალებს. ფაილების ჟურნალები გენერირდება დროის მიხედვით, როდესაც ინსტალირების თითოეული მოქმედება იქნა გამოძახილი. ამ ფაილში ნაჩვენებია ქმედებების შესრულების თანმიმდევრობა და მათი დამოკიდებულებები.

თუ ინსტალირებისას განხორციელდა შეცდომა, იგი რეგისტრირებული იქნება ამ ფაილის ბოლოში, რომელიც, ასევე, შეიძლება მოდიებულ იქნეს საკვანძო სიტყვებით error ან exception.

# MS SQL Server-ის ინსტალირების ჟურნალი

## MSI Log Files

ამ ფაილებში არის ინფორმაცია პაკეტების პროცესის ინსტალირების შესახებ. ისინი გენერირდებიან MSIEXEC-ის მიერ გარკვეული პაკეტების ინსტალირების დროს. MSI ჟურნალების ფაილების ტიპებია:

**<Feature>\_<Architecture>\_<Interaction>.log**

**<Feature>\_<Architecture>\_<Language>\_<Interaction>.log**

**<Feature>\_<Architecture>\_<Interaction>\_<workflow>.log**

ფაილის ბოლოს არის ჯამური ინფორმაცია შესრულებაზე, რომელშიც მითითებულია საერთო მდგომარეობა (წარმატება ან წარუმატებლობა) და თვისებები. MSI ფაილში შეცდომის მოძიება უნდა განხორციელდეს მნიშვნელობით "value 3" და ინახოს მასთან მდებარე ტექსტი.

## ConfigurationFile.ini

კონფიგურაციის ფაილი შეიცავს ინსტალირების შემავალ პარამეტრებს, რომელიც შეიძლება გამოყენებულ იქნეს განმეორებით ინსტალირებისას (გარდა ანგარიშების პაროლებისა, PID-ისა და ზოგიერთ პარამეტრებისა, რომლებიც კონფიგურაციის ფაილში არ ინახება).

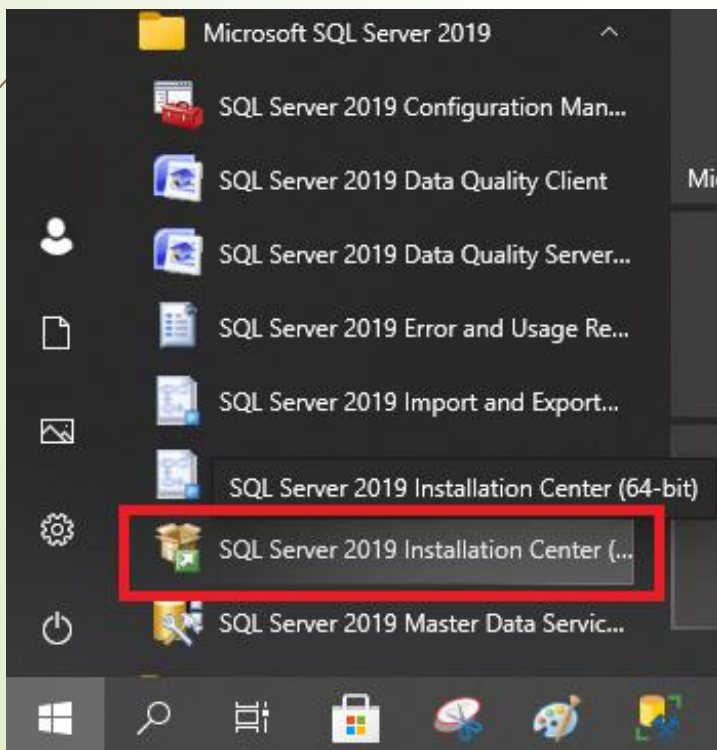
## SystemConfigurationCheck\_Report.htm

ყველა წესის მოკლე აღწერა და შესრულების სტატუსი.



# MS SQL Server-ის ინსტალირების ცენტრი

კომპიუტერში ინსტალირებულ SQL Server-ის ინსტალირების ცენტრის (დიალოგური ფანჯრის) გამოსაძახებლად მომხმარებელმა Start მენიუს გამოყენებით უნდა შევიდეს Microsoft SQL Server <ვერსიის სახელი> (ჩვენ შემთხვევაში Microsoft SQL Server 2019) და ახლად ჩამოშლილ სიაში უნდა გაააქტიუროს სტრიქონი SQL Server <ვერსიის სახელი> Installation Center (ჩვენ შემთხვევაში SQL Server 2019 Installation Center), რის შემდეგაც ეკრანზე გამოისახება ინსტალირების სტანდარტული დიალოგური ფანჯარა, რომელშიც შესაძლებელია SQL Server-ის ყველა ინსტალირების ფუნქციის გაშვება.

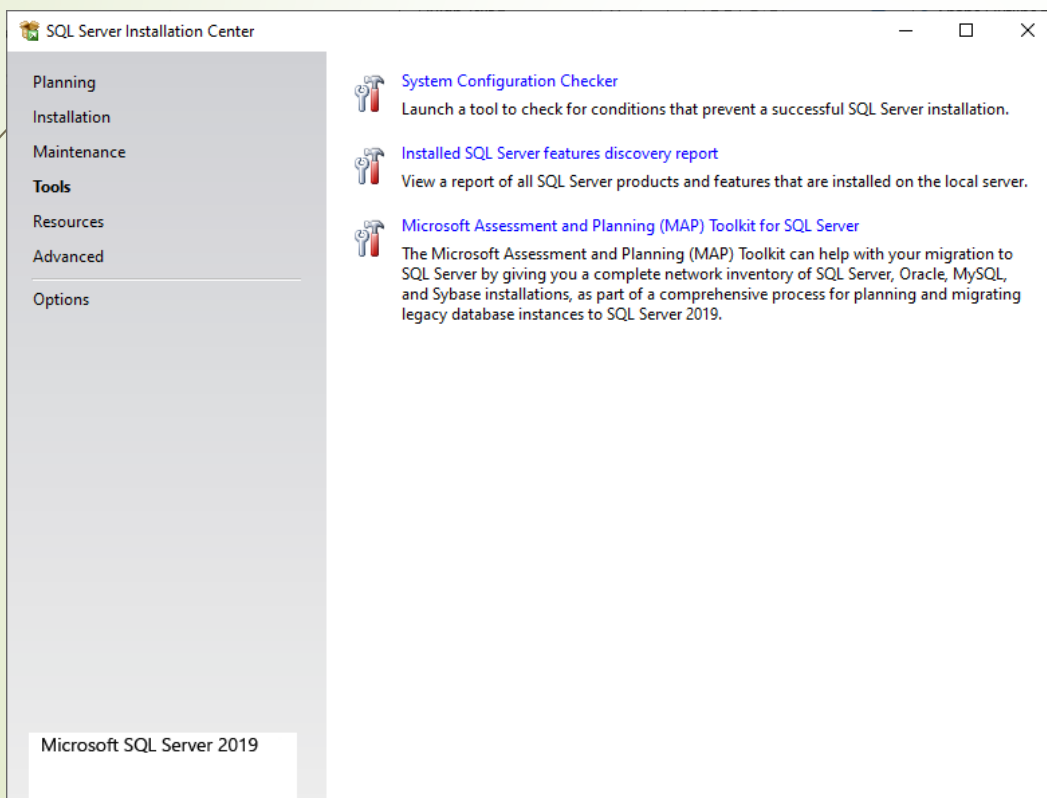




# MS SQL Server-ის ინსტალირების შემოწმება

SQL Server-ის ინსტალირების ცენტრის დიალოგურ ფანჯარაში მარცხენა მიდამოში მდებარე მენიუში უნდა მოინიშნოს მეოთხე სტრიქონი Tools, რის შემდეგაც მარჯვენა მიდამოში გააქტიურდეს მეორე სტრიქონი „ინსტალირებული SQL Server-ის კომპონენტების აღმოჩენის ანგარიში“ (Installed SQL Server features discovery report), რის შემდეგაც ანგარიში ინახება საქალაქდებში:

%ProgramFiles %\MicrosoftSQL Server\nnn\Setup Bootstrap\Log\<ბოლო\_სესია>



ზემოაღნიშნული ანგარიში შესაძლებელია შეიქნას ასევე PowerShell-ის ბრძანებათა სტრიქონის გამოყენებით, რისთვისაც მომხმარებელმა უნდა გაუშვას ბრძანება:

**Setup.exe /Action = RunDiscovery**

თუ დამატებით გამოყენებულ იქნება პარამეტრი „/q“, შედეგი არ იქნება ასახული, მაგრამ ფაილი მაინც შეიქმნება ზემოაღნიშნულ საქალაქდებში.

# 3 LEQC

# MS SQL Server-ის ინსტალირება Windows Server 2016 Core და Windows Server 2019 Core-ში

Windows Server 2016 Core და Windows Server 2019 Core-ში ინსტალირებისას პარამეტრების მითითებებს აქვთ შემდეგი დანიშნულება:

- **/Q** - სრული ჩუმი რეჟიმი შეტყობინებების გამოტანის გარეშე;
- **/QS** - მარტივი რეჟიმი შეტყობინებების გამოტანის გარეშე;
- **/IACCEPTSQLSERVERLICENSETERMS** - ლიცენზირების პირობებთან დათანხმება;
- **/FEATURES** - დამატებითი პარამეტრების მისათითებლად:
  - **/FEATURES=SQLENGINE** - დაინსტალირდება მხოლოდ Database Engine კომპონენტი;
  - **/FEATURES=REPLICATION** - დაინსტალირდება რეპლიკაციის კომპონენტი Database Engine-თან ერთად;
  - **/FEATURES=FULLTEXT** - დაინსტალირდება Full-Text Search კომპონენტი Database Engine-თან ერთად;
  - **/FEATURES=AS** - დაინსტალირდება Analysis Services-ის სამსახურის ყველა კომპონენტი;
  - **/FEATURES=IS** - დაინსტალირდება Integration Services-ის სამსახურის ყველა კომპონენტი;
  - **/FEATURES=CONN** - დაინსტალირდება მონაცემებთან მიერთების კომპონენტი Database Engine კომპონენტთან ერთად.

# MS SQL Server-ის ინსტალირება Windows Server 2016 Core და Windows Server 2019 Core-ში

მოვიყვანოთ Windows Server 2016 Core-ში ან Windows Server 2019 Core-ში ინსტალირებისას პარამეტრების მითითებების

მაგალითი:

```
Setup.exe /qs  
/ACTION=Install  
/FEATURES=SQLEngine, Replication  
/INSTANCENAME=MSSQLSERVER  
/SQLSVCACCOUNT="<დომენის_სახელი\მომხმარებლის_სახელი>"  
/SQLSVCPASSWORD="<მძლავრი_პაროლი>"  
/SQLSYSADMINACCOUNTS="< დომენის_სახელი\მომხმარებლის_სახელი >"  
/AGTSVCACCOUNT="NT AUTHORITY\Network Service"  
/TCPENABLED=1  
/IACCEPTSQLSERVERLICENSETERMS
```



# MS SQL Server-ის ინსტალირება Windows Server 2016 Core და Windows Server 2019 Core-ში

ასევე შესაძლებელია ინსტალირება განხორციელდეს კონფიგურაციის ფაილის (ტექსტური ფაილის INI გაფართოების) გამოყენებით, რომელიც შეიცავს პარამეტრებს (ანუ წყვილებს „სახელი-მნიშვნელობა“) და კომენტარებს აღწერით, რომლის დროსაც ინსტალირების განსახორციელებლად ბრძანების სტრიქონში უნდა დაიწეროს:

**Setup.exe /QS /ConfigurationFile=MyConfigurationFile.INI**

მოვიყვანოთ კონფიგურაციის ფაილით Database Engine-ს კომპონენტის ინსტალირების

მაგალითი:

**[OPTIONS]**

**ACTION="Install"**

**FEATURES=SQLENGINE**

**INSTANCENAME="MSSQLSERVER"**

**INSTANCEID="MSSQLSERVER"**

**SQLSVCACCOUNT="NT Service\MSSQLSERVER"**

**SQLSYSADMINACCOUNTS="\< დომენის\_სახელი\მომხმარებლის\_სახელი >"**

**IAcceptSQLServerLicenseTerms="True"**

# MS SQL Server-ის ინსტალირება Windows Server 2016 Core და Windows Server 2019 Core-ში

ასევე, მოვიყვანოთ კონფიგურაციის ფაილით მონაცემებთან მიერთების კომპონენტის ინსტალირების

მაგალითი:

```
[OPTIONS]  
ACTION="Install"  
FEATURES=Conn  
IAcceptSQLServerLicenseTerms="True"
```

იმისათვის, რომ დაშვებულ იქნეს გარე მიერთებები SQL Server-ზე, უნდა განხორციელდეს შემდეგი ინსტრუქციები:

```
EXEC sys.sp_configure N'remote access', N'1'  
GO  
RECONFIGURE WITH OVERRIDE  
GO
```

# MS SQL Server-ის ინსტალირება Windows Server 2016 Core და Windows Server 2019 Core-ში

SQL Server-ში ბრაუზერის ჩასართავად (ავტომატურად იგი ინსტალირებისას გამორთულია), უნდა განხორციელდეს შემდეგი ინსტრუქცია:

**Set-service sqlbrowser -StartupType Auto**

რის შემდეგაც ბრაუზერის გასაშვებად საჭიროა შემდეგი ინსტრუქცია:

**Start-service sqlbrowser**

SQL Server-ში TCP/IP ოქმის მხარდაჭერის ჩასართავად, უნდა განხორციელდეს შემდეგი:

- PowerShell-ში გაუშვათ Import-Module SQLPS;
- SQL Server-ის PowerShell-ში გაუშვათ შემდეგი ინსტრუქცია:

```
$smo = 'Microsoft.SqlServer.Management.Smo.'
$wmi = new-object ($smo + 'Wmi.ManagedComputer')
$uri = "ManagedComputer[@Name='" + (get-item env:\computername).Value +
    "']/ServerInstance[@Name='MSSQLSERVER']/ServerProtocol[@Name='Tcp'],,
$Tcp = $wmi.GetSmoObject($uri)
$Tcp.IsEnabled = $true
$Tcp.Alter()
$Tcp
```

# SQL Server-ის რესურსების შეზღუდვები

## მონაცემთა ბაზის ძრავის ობიექტები

პაკეტის ზომა - 65536 ბაიტი (ქსელის პაკეტის ზომა).

სტრიქონის სიგრძე, რომელიც შეიცავს **Transact-SQL** ინსტრუქციებს (პაკეტის ზომა) - 65536 ბაიტი (ქსელის პაკეტის ზომა).

ორივე შემთხვევაში ქსელის პაკეტის ზომა არის ცხრილის მონაცემთა ნაკადის (TDS) პაკეტების ზომა, რომელიც გამოიყენება პროგრამებსა და მონაცემთა ბაზის ძრავას შორის ურთიერთობისათვის. ნაგულისხმევი პაკეტის ზომაა 4 KB და კონტროლდება ქსელის პაკეტის ზომის კონფიგურაციის პარამეტრით.

მოკლე სტრიქონის სვეტი - 8000 ბაიტი.

**GROUP BY** და **ORDER BY** – 8060 ბაიტი.

ინდექსის გასაღები - 900 ბაიტი კლასტერული ინდექსისთვის, 1700 ბაიტი არაკლასტირებული ინდექსისთვის.

ინდექსის გასაღები მეხსიერებაში ოპტიმიზირებულ ცხრილებისთვის - 2500 ბაიტი არაკლასტირებული ინდექსისთვის.

(გაგრძელება შემდეგ სლაიდზე)



# SQL Server-ის რესურსების შეზღუდვები

73

გარე გასაღების ზომა - 900 ბაიტი.

პირველადი გასაღების ზომა - 900 ბაიტი.

სტრიქონის ზომა - 8 060 ბაიტი.

სტრიქონის ზომა მეხსიერებაში ოპტიმიზირებულ ცხრილებისთვის - 8 600 ბაიტი.

შენახული პროცედურის ტექსტის ზომა - პაკეტის ზომაზე ნაკლები ან 250 მბ.

**varchar(max)** , **varbinary(max)**, **xml**, **text**, ან **image** სვეტის ზომა -  $2^{31}-1$  ბაიტი.

**ntext** ან **nvarchar(max)** სვეტის ზომა -  $2^{30}-1$  სიმბოლო.

კლასტერული ინდექსების რაოდენობა თითო ცხრილში - 1.

სვეტების რაოდენობა **GROUP BY**-ში ან **ORDER BY**-ში - შეზღუდულია მხოლოდ ბაიტებით.

სვეტების ან გამოსახულებების რაოდენობა **GROUP BY WITH CUBE** ან **WITH ROLLUP** ინსტრუქციებში - 10.

ინდექსის გასაღებში სვეტების რაოდენობა - 32.

გარე ან პირველად გასაღებში სვეტების რაოდენობა - 32.

(გაგრძელება შემდეგ სლაიდზე)

# SQL Server-ის რესურსების შეზღუდვები

74

**INSERT** ინსტრუქციაში სვეტების რაოდენობა - 4 096.

**SELECT** ინსტრუქციაში სვეტების რაოდენობა - 4 096.

**UPDATE** ინსტრუქციაში სვეტების რაოდენობა - 4 096.

ცხრილში სვეტების რაოდენობა - 1 024.

წარმოდგენაში სვეტების რაოდენობა - 1 024.

მიერთებების რაოდენობა ერთ კლიენტთან - მიერთებების კონფიგურირების მაქსიმალური მნიშვნელობა.

მონაცემთა ბაზის ზომა - 524 272 ტერაბაიტი.

**SQL Server** ერთ ეგზემპლარში მონაცემთა ბაზების რაოდენობა - 32 767.

ფაილური ჯგუფების რაოდენობა ერთ მონაცემთა ბაზაში - 32 767.

მეხსიერებისათვის ოპტიმიზირებული ფაილური ჯგუფების რაოდენობა მონაცემებისთვის ერთ მონაცემთა ბაზაზე - 1.

ფაილების რაოდენობა ერთ მონაცემთა ბაზაში - 32 767.

(გაგრძელება შემდეგ სლაიდზე)

# SQL Server-ის რესურსების შეზღუდვები

მონაცემთა ფაილის ზომა - 16 ტერაბაიტი.

ჟურნალის ფაილის ზომა - 2 ტერაბაიტი.

გარე გასაღების ბმულების რაოდენობა ერთ ცხრილისათვის - გამავალი 253, ხოლო შემომავალი 10 000.

იდენტიფიკატორის სიგრძის ზომა - 128 სიმბოლო.

ეგზემპლიარების რაოდენობა თითო კომპიუტერზე - 50.

ინდექსების რაოდენობა მეხსიერებისათვის ოპტიმიზირებულ ცხრილში - 999.

ბლოკირებების რაოდენობა ერთ მიერთებაზე - მაქსიმალური ბლოკირების რიცხვი სერვერზე.

ბლოკირებების რაოდენობა ერთ SQL Server-ის ეგზემპლიარზე - შეზღუდულია მხოლოდ მეხსიერების მოცულობით.

ჩადგმული შენახული პროცედურების დონეების რაოდენობა - 32.

ჩადგმული მოთხოვნების დონეების რაოდენობა - 32.

(გაგრძელება შემდეგ სლაიდზე)

# SQL Server-ის რესურსების შეზღუდვები

76

ჩადგმული ტრანზაქციების დონეების რაოდენობა - 4 294 967 296.

ჩადგმული ტრიგერების დონეების რაოდენობა - 32.

ცხრილში არაკლასტირებული ინდექსების რაოდენობა - 999.

შენახული პროცედურის პარამეტრების რაოდენობა - 2100.

მომხმარებლის ფუნქციის პარამეტრების რაოდენობა - 2100.

ცხრილში მითითებების რაოდენობა - 253.

ცხრილში სტრიქონების რაოდენობა - შეზღუდულია ხელმისაწვდომი მეხსიერებით.

მონაცემთა ბაზაში ცხრილების, წარმოდგენების, შენახული პროცედურების, მომხმარებლის ფუნქციების, ტრიგერების, წესების, ნაგულისხმევ მნიშვნელობებისა და შეზღუდვების ჯამური რაოდენობა - 2147483647.

მომხმარებელთა მიერთებების რაოდენობა - 32767.

XML ინდექსების რაოდენობა - 249.



# SQL Server-ის რესურსების შეზღუდვები

77

## SQL Server-ის უტილიტების ობიექტები

SQL Server-ის უტილიტებში კომპიუტერების რაოდენობა (ფიზიკური ან ვირტუალური მანქანები) რაოდენობა - 100.

SQL Server-ის ეგზემპლიარების რაოდენობა კომპიუტერში - 5.

SQL Server-ის უტილიტებში ეგზემპლიარების რაოდენობა - 200.

SQL Server-ის ეგზემპლიარში მომხმარებლის მონაცემთა ბაზების რაოდენობა - 50.

SQL Server-ის უტილიტებში მომხმარებლის მონაცემთა ბაზების რაოდენობა - 1 000.

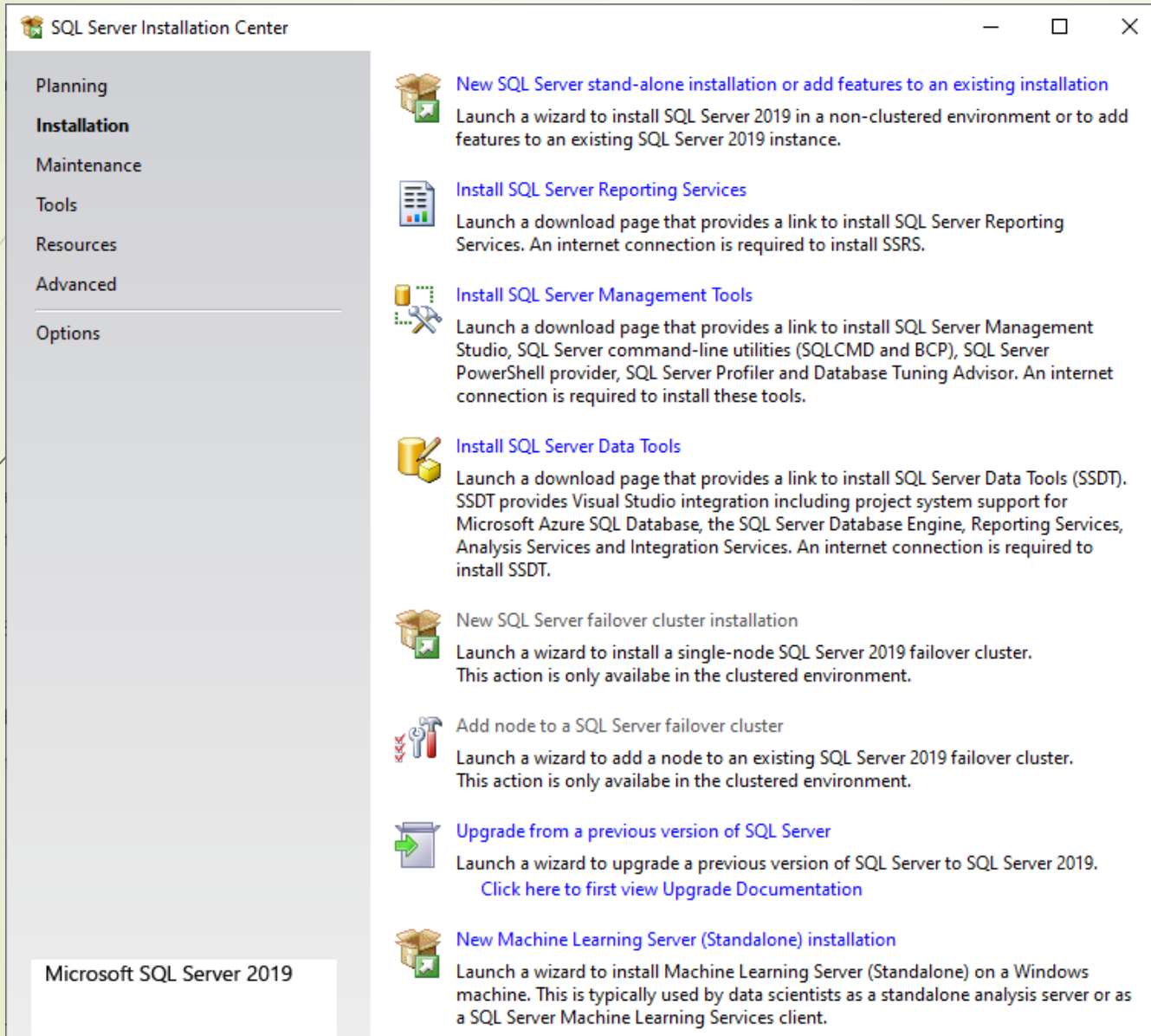
ფაილური ჯგუფების რაოდენობა მონაცემთა ბაზაში - 1.

მონაცემთა ფაილების რაოდენობა ფაილურ ჯგუფში - 1.

ჟურნალების ფაილების რაოდენობა მონაცემთა ბაზაში - 1.

ტომების რაოდენობა კომპიუტერში - 3.

# MS SQL Server-ის განახლება



SQL Server-ის განახლება (Update) ავტომატურად ხორციელდება Windows-ის განახლებისას, მაგრამ წინა ვერსიის უფრო მაღალ ვერსიამდე განახლება (Upgrade) შესაძლებელია განხორციელდეს ინსტალირების ცენტრის დიალოგური ფანჯრის გამოყენებით. ამისათვის მარცხენა მიდამოში მდებარე მენიუში უნდა მოინიშნოს მეორე სტრიქონი Installation, რის შემდეგაც მარჯვენა მიდამოში გააქტიურდეს ქვემოდან მეორე სტრიქონი Upgrade from previous versions of SQL Server.

# MS SQL Server-ის განახლება

პროდუქტის გასაღების (Product Key) გვერდზე მომხმარებელმა უნდა მიუთითოს განახლება ხორციელდება SQL Server-ის უფასო გამოცემამდე, თუ მას გააჩნია PID გასაღები პროდუქტის მუშა ვერსიისათვის.

ლიცენზიის პირობების (License Terms) გვერდზე მომხმარებელმა უნდა გადახედოს სალიცენზიო ხელშეკრულებას და თუ თანახმა არის, მონიშნოს თანხმობა (I accept the license terms).

გლობალური წესების (Global Rules) ფანჯარაში, ინსტალირების პროცედურა ავტომატურად გადავა პროდუქტის განახლებების ფანჯარაში, თუ არ გამოიკვეთა შეცდომები წესებში.

Microsoft-ის განახლების (Microsoft Update) გვერდი გამოჩნდება, თუ Windows-ში Control Panel\All Control Panel Items\Windows Update\Change settings-ში მონიშნულია Microsoft Update ველი.

პროდუქტის განახლებების (Product Updates) გვერდზე ნაჩვენები იქნება უახლესი ხელმისაწვდომი SQL Server პროდუქტის განახლებები. თუ პროდუქტის განახლებები არ არის აღმოჩენილი, გვერდი არ გამოისახება და ავტომატურად გადავა ინსტალირების ფაილების (Install Setup Files) გვერდზე.

(გაგრძელება შემდეგ სლაიდზე)



# MS SQL Server-ის განახლება

ინსტალირების ფაილების (Install Setup Files) გვერდზე, შემოთავაზებულ იქნება ფაილების გადმოტვირთვის პროგრესი, მოპოვებისა და ინსტალაციისთვის.

განახლების წესების (Upgrade Rules) ფანჯარაში, დაყენების პროცედურა ავტომატურად გადავა ეგზემპლიარის შერჩევის (Select Instance) ფანჯარაში, თუ არ გამოისახა წესის შეცდომები.

ეგზემპლიარის შერჩევის (Select Instance) გვერდზე მიუთითეთ SQL სერვერის ეგზემპლიარი, რომელიც უნდა განახლდეს. მხოლოდ გაზიარებული კომპონენტების განახლებისათვის უნდა შეირჩეს Upgrade shared features only.

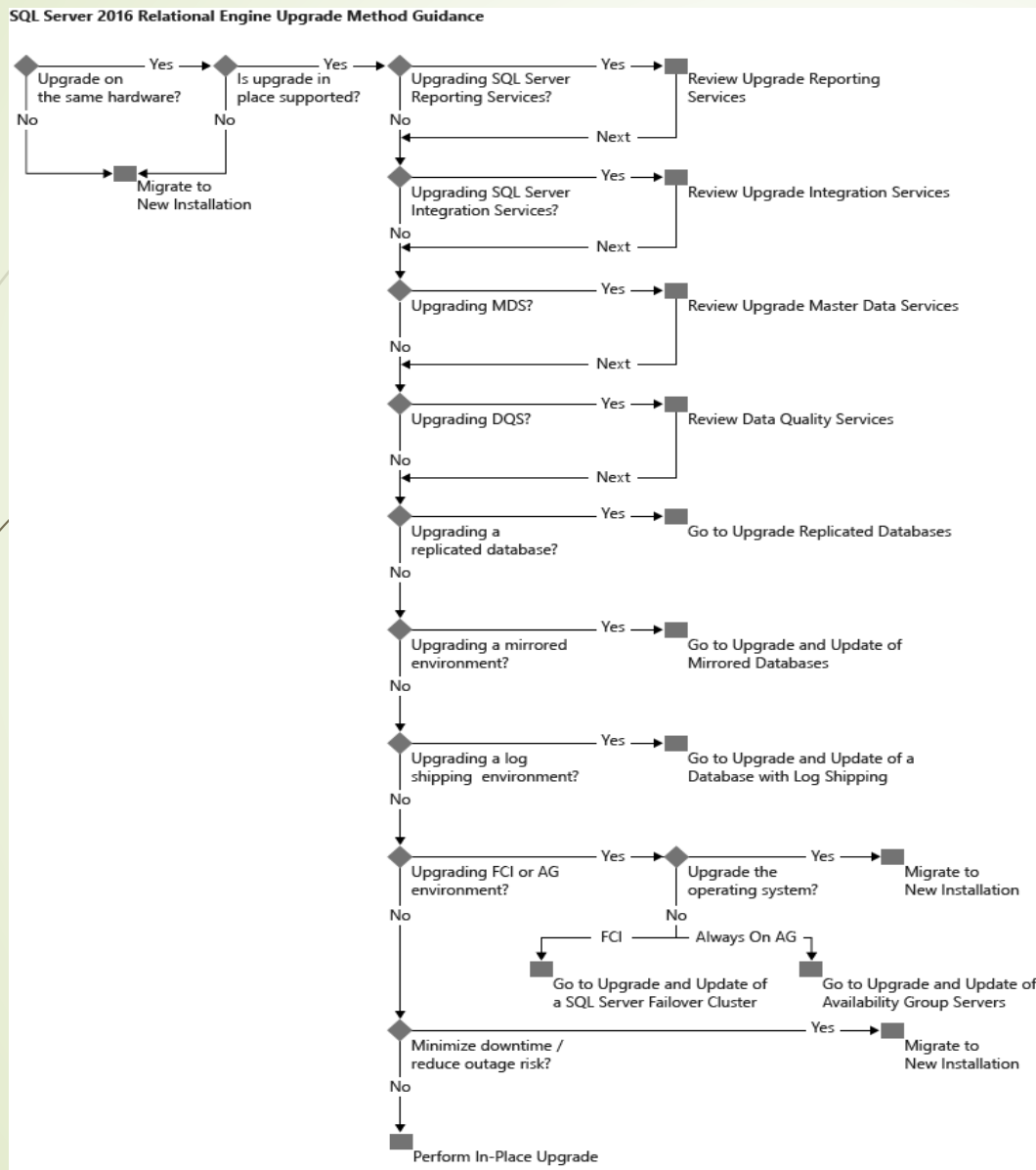
კომპონენტების შერჩევის (Select Features) გვერდზე, განახლებულ იქნება წინასწარ შერჩეული კომპონენტები. თითოეული შერჩეული კომპონენტის ჯგუფის აღწერა გამოჩნდება მარჯვენა მიდამოში.

ეგზემპლიარის კონფიგურირების (Instance Configuration) გვერდზე უნდა იქნეს მითითებული SQL Server-ის ეგზემპლიარის ID, რომელიც წარმოადგენს ეგზემპლიარის სახელს. SQL Server-ის ყველა სერვისული პაკეტი და განახლება ვრცელდება SQL Server-ის ეგზემპლიარის ყველა კომპონენტზე.



# MS SQL Server-ის Database Engine-ს განახლება

81



არსებობს SQL Server-ის წინა ვერსიების მონაცემთა ბაზის ძრავის (Database Engine) განახლების რამდენიმე მეთოდი, რათა მაქსიმალურად იქნეს შემცირებული SQL Server-ის გათიშვის დრო და რისკი:

- განახლება ადგილზე;
- მიგრაცია ახალ ინსტალაციაზე;
- მიმდევრობითი განახლება.

სლაიდზე მოყვანილი დიაგრამა ასახავს სამივე მიდგომას და დაეხმარება მომხმარებელს სწორი მიდგომის შერჩევაში.

# MS SQL Server-ის Database Engine-ს განახლება

82

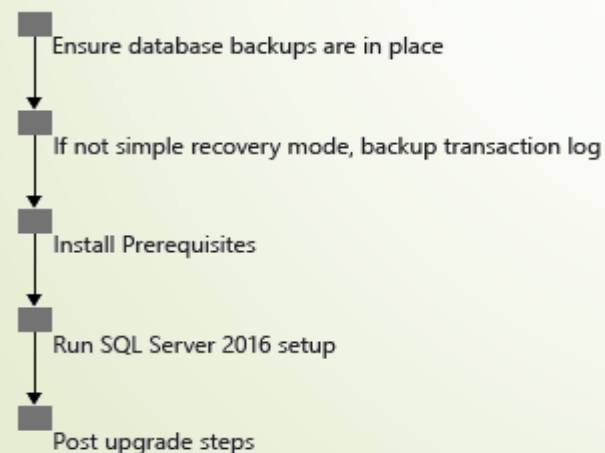
განვიხილოთ სამივე მიდგომა:

განახლება ადგილზე

ამ შემთხვევაში SQL Server-ის განახლება ხორციელდება იგივე კომპიუტერში და იგივე ოპერაციულ გარემოში - განახლებას ექვემდებარება მხოლოდ SQL Server-ი, რომელიც არსებული SQL Server-ის ბიტების შეცვლის ახალი SQL Server-ის ბიტებით და შემდგომ ახორციელებს სისტემისა და მომხმარებლის მონაცემთა ბაზების განახლებას.

განახლების ადგილზე მიდგომა არის ყველაზე მარტივი მეთოდი და მოითხოვს გარკვეული დროის უმოქმედობას, უფრო მეტი დრო სჭირდება მტყუნებისას უკან დაბრუნებას. გარდა ამისა ეს მეთოდი არ არის მხარდაჭერილი ყველა სცენარისათვის. ძირითად, ეს მიდგომა გამოიყენება იმ შემთხვევაში, როდესაც წარმოების გარემო არ არის კრიტიკული და მან შეუძლია გაუძლოს გათიშვის გარკვეულ დროს.

SQL Server 2016 Relational Engine In-Place Upgrade Method



SQL Server-ის განახლებისას წინა ვერსია გადაიწერება და აღარ იარსებებს კომპიუტერში, ამიტომ განახლებამდე უნდა შეიქმნას SQL Server-ის მონაცემთა ბაზებისა და სხვა ობიექტების სარეზერვო კოპია.

მოცემული დიაგრამა არის მონაცემთა ბაზის ძრავის ადგილზე განახლებისთვის სქემა.

# MS SQL Server-ის Database Engine-ს განახლება

83

## მიგრაცია ახალ გარემოში

ამ მიდგომით შენარჩუნებული იქნება არსებული SQL Server-ის სისტემა და ხორციელდება ახალ ოპერაციულ გარემოში ან/და ახალ კომპიუტერში ახალი SQL Server-ის შექმნა. SQL Server-ის ახალ გარემოში დაყენების შემდეგ, ხორციელდება მთელი რიგი ნაბიჯები ახალი გარემოს მოსამზადებლად, რათა გახდეს შესაძლებელი არსებული მომხმარებელთა მონაცემთა ბაზების მიგრაცია ახალ გარემოში და მინიმუმამდე იქნეს დაყვანილი გათიშვის დრო. ამ მიდგომისას მიგრაციას განიცდიან შემდეგი კომპონენტები:

- **სისტემური ობიექტები:** როგორც წესი, სამომხმარებლო პროგრამა არის დამოკიდებული როგორც სისტემურ master და msdb, ასევე სამომხმარებლო მონაცემთა ბაზებზე. ყველაფერი, რაც იქნა შენახული სამომხმარებლო მონაცემთა ბაზის გარეთ და საჭიროა მისი სწორი ფუნქციონირებისათვის, ხელმისაწვდომი უნდა იყოს ახალ SQL Server-ზეც. მაგალითად:
  - წვდომის პარამეტრები, რომლებიც ინახება master სისტემურ მონაცემთა ბაზაში;
  - SQL Server აგენტის ფუნქციონირება, რომელთა მეტამონაცემები ინახება msdb სისტემურ მონაცემთა ბაზაში;
  - სერვერის დონის ტრიგერები, რომლებიც ინახება master სისტემურ მონაცემთა ბაზაში.

ყველა ამ შემთხვევაში ისინი ხელახლა უნდა შეიქმნან ახალ SQL Server-ში.

(გაგრძელება შემდეგ სლაიდზე)



# MS SQL Server-ის Database Engine-ს განახლება

- **MSDB** მონაცემთა ბაზაში შენახული **Integration Services** პაკეტები: თუ პაკეტები შენახულია MSDB მონაცემთა ბაზაში, მაშინ საჭიროა ამ პაკეტების სკრიპტში ჩაწერა dtutil უტილიტების გამოყენებით (დაწვრილებითი ცნობები არის ვებ-გვერდზე <https://docs.microsoft.com/en-us/sql/integration-services/dtutil-utility?view=sql-server-ver15>), რის შემდეგაც პაკეტები უნდა განახლდნენ ახალი SQL Server-ის ვერსიამდე და შემდგომ უნდა გაიშალოს ახალ SQL Server-ში.
- **Reporting Services-ის** სერვისების დაშიფვრის გასაღებები: Reporting Services-ის კონფიგურაციის მნიშვნელოვანი ნაწილია სიმეტრიული გასაღების სარეზერვო ასლის შექმნა, რომელიც გამოიყენება კონფიდენციალური ინფორმაციის დასაშიფრად. გასაღების სარეზერვო ასლი საჭიროა მრავალი რუტინული ოპერაციისთვის და იძლევა საშუალებას გამოყენებულ იქნეს არსებული სერვერის მონაცემთა ბაზა ახალ SQL Server-ში.

რადგან ახალ SQL Server-ს გააჩნია იგივე სისტემური ობიექტები, როგორც ძველს, ამ მიდგომის გამოყენებისას სამომხმარებლო მონაცემთა ბაზებს მიგრაციის უმოქმედობის დრო მინიმუმამდე არის დაყვანილი. მონაცემთა ბაზების მიგრაციის განხორციელება შესაძლებელია სარეზერვო ასლის შექმნისა და აღდგენის გზით, ან LUN-ის SAN გარემოში გადაგზავნით.

(გაგრძელება შემდეგ სლაიდზე)



# MS SQL Server-ის Database Engine-ს განახლება

მომხმარებლის მონაცემთა ბაზის მიგრაციის შემდეგ ერთერთი გზა მომხმარებლების ახალ თქვენ SQL Server-თან მიერთების არის DNS ჩანაწერებში სერვერის სახელის გადარქმევა და მიერთების სტრიქონების შეცვლა.

ამ მიდგომის გამოყენება მიზანშეწონილია თუ:

- SQL Server-ის ინსტალირება უნდა განხორციელდეს არამხარდამჭერ ოპერაციულ გარემოში (მაგ. ძველი SQL Server-ი არის x86 და გადატანილ უნდა იქნეს Windows Server 2016-ზე, რომელიც არის x64;
- SQL Server-ი აკეთებს მიგრაციას ახალ აპარატურაზე და/ან ახალ ოპერაციულ გარემოში;
- არაკლასტერული SQL Server-ი აკეთებს მიგრაციას კლასტერულ SQL Server-ში.
- SQL Server 2005 აკეთებს მიგრაციას SQL Server 2016-ში (13.x), რადგან იგი უკვე აღარ არის მხარდაჭერილი.

ახალ გარემოში მიგრაციის მოქმედებები განსხვავდება იმისდა მიხედვით გამოყენებულ იქნება მიერთებული საცავი თუ SAN საცავი

(გაგრძელება შემდეგ სლაიდზე)

# MS SQL Server-ის Database Engine-ს განახლება

86

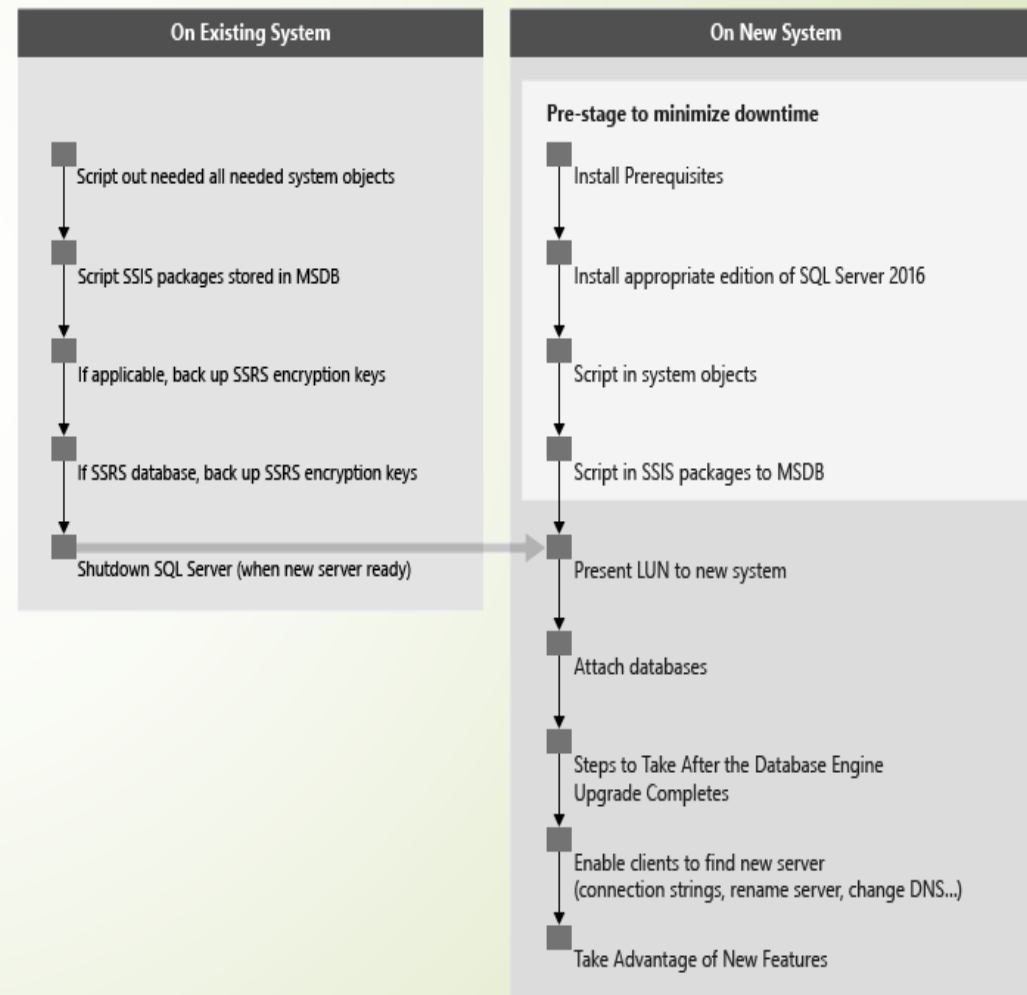
მიერთებული საცავის გარემო:

SAN საცავის გარემო:

SQL Server 2016 Relational Engine  
New Installation Upgrade Method Using Backup and Restore with Attached Storage



SQL Server 2016 Relational Engine  
New Installation Upgrade Method Using Detach and Attach with SAN Storage



# MS SQL Server-ის Database Engine-ს განახლება

87

## მიმდევრობითი განახლება

მიმდევრობითი განახლება საჭიროა როდესაც უნდა განახლდეს SQL Server-ის რამდენიმე ეგზემპლარი, რომლებიც დროის უმოქმედობისა და რისკების შემცირებისათვის და გარემოს ფუნქციონირების შესანარჩუნებლად ხორციელდება. ანუ ამ მიდგომისას ხორციელდება SQL Server-ის რამდენიმე ეგზემპლარის განახლება კონკრეტული თანმიმდევრობით - ხორციელდება SQL Server-ის ყოველი კონკრეტული ეგზემპლარის განახლება ან ახალი ინსტალირება.

# MS SQL Server-ის თავსებადობის დონის ცვლილება

SQL Server 2016-ში (13.x) და უფრო მაღალ ვერსიებში ზოგიერთი ცვლილება მხოლოდ მონაცემთა ბაზის თავსებადობის დონის შეცვლის შემდეგ არის შესაძლებელი. ეს გაკეთდა რამდენიმე მიზეზის გამო:

- შეუძლებელია SQL Server-ის მონაცემთა ბაზის თავსებადობის დონის დაქვეითება და თუ მომხმარებელს სურს ძველ ვერსიაზე დაბრუნება, იგი ამას ვეღარ გააკეთებდა;
- იგივე პრობლემა შეიძლება შეიქმნას სისტემის რაღაც ნაწილის განახლებისას წარმატების შემთხვევაში, შეიძლება მიუღებელი რეგრესია გამოიწვიოს სხვებისთვის.

მოთხოვნების ახალი დამამუშავებლის ფუნქციონირების გასააქტიურებლად განახლების პროცესი დაკავშირებულია პროდუქტის გამოშვების შემდგომ მომსახურების მოდელთან. ზოგიერთი ამ გამოსწორებებიდან გამოშვებულ იქნა ნომრით 4199, რომელიც მოულოდნელი რეგრესიის რისკის გარეშე შესაძლებელია დაყენდეს. დაწყებული SQL Server 2016-დან (13.x) აღარ არის საჭირო 4199 განახლება.

SQL Server-ის ძველი ვერსიიდან SQL Server 2014 (12.x) ან უფრო ახალ ვერსიაზე გადასვლისას მონაცემთა ბაზის თავსებადობის დონის უახლეს ვერსიამდე გადასვლისას, დატვირთვამ შესაძლებელია განიცადოს რეგრესია. ეს უფრო მცირე ხარისხით ასევე შესაძლებელია SQL Server 2014 (12.x)-ის ნებისმიერ ახალ ვერსიაზე განახლებისას.

(გაგრძელება შემდეგ სლაიდზე)



# MS SQL Server-ის მბ-ს განახლება QTA-ს გამოყენებით

SQL Server 2014 (12.x)-დან დაწყებული მოთხოვნების ოპტიმიზების ყველა ცვლილება დაკავშირებულია მონაცემთა ბაზის თავსებადობის ბოლო დონესთან, ამიტომ შესრულების გეგმები განახლების მომენტში კი არ იცვლება, არამედ როდესაც მომხმარებელი შეცვლის მონაცემთა ბაზის პარამეტრს COMPATIBILITY\_LEVEL უახლეს ხელმისაწვდომამდე.

განახლებების ეს კონტროლი კიდევ უფრო გაუმჯობესდა SQL Server 2017 (14.x)-დან, რომელშიც დაინერგა ავტომატური რეგულირება და იგი საშუალებას იძლევა ბოლო ბიჯის ავტომატიზებას.

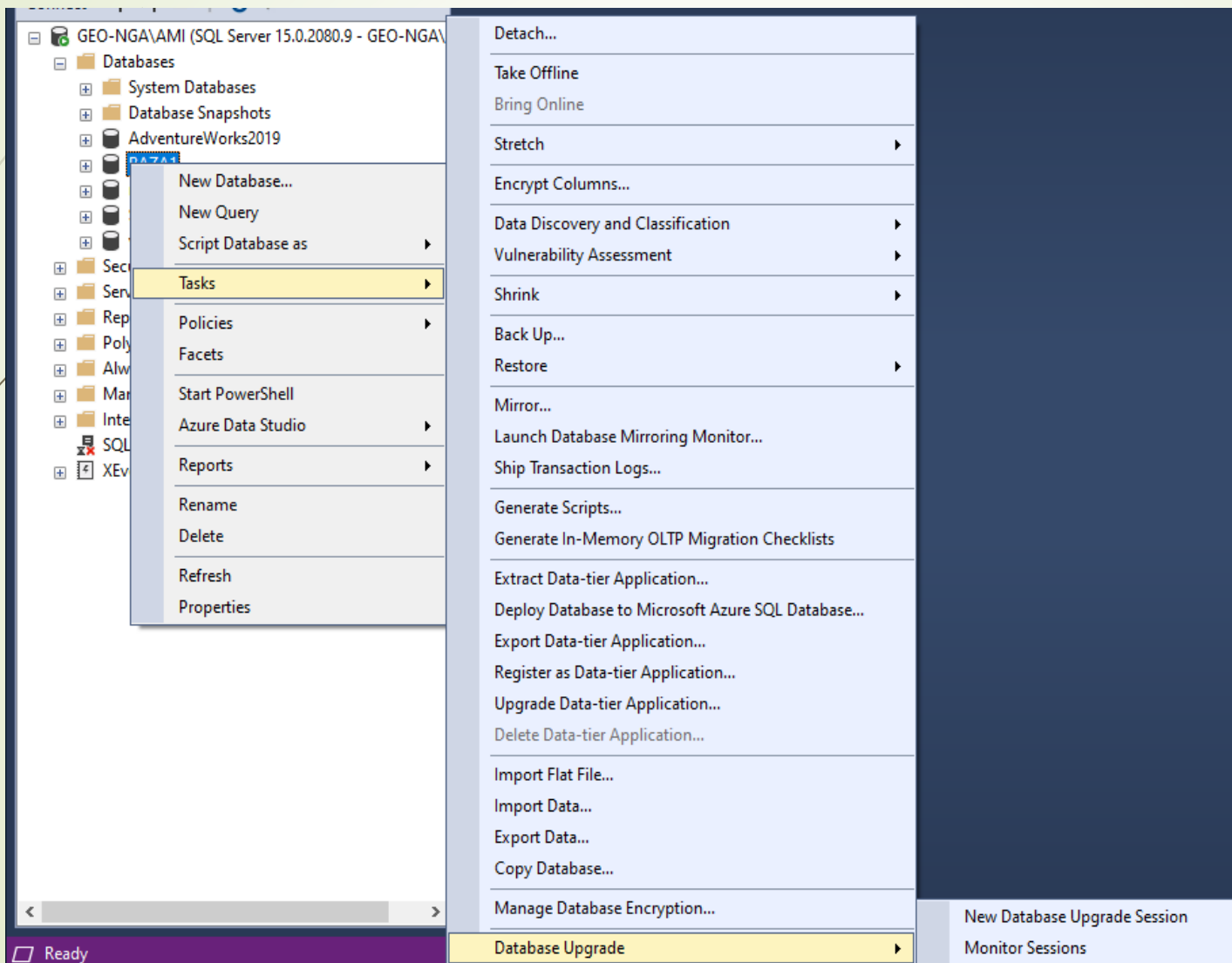
SQL Server Management Studio v18-დან დაწყებული, მომხმარებლებს აქვს შესაძლებლობა განახორციელოს განახლება მონაცემთა ბაზების განახლება „მოთხოვნის რეგულირების ასისტენტის“ (Query Tuning Assistant - QTA) გამოყენებით.

QTA არის სესიაზე დაფუძნებული კომპონენტი, რომელიც ინახავს სესიის მდგომარეობას მომხმარებლის მონაცემთა ბაზის (რომელშიც დაიწყო სეანსი) msqta სქემაში. დროთა განმავლობაში ერთ მონაცემთა ბაზაში შესაძლებელია შეიქმნას მრავალი QTA სესია, მაგრამ ნებისმიერი მონაცემთა ბაზისთვის შეიძლება არსებობდეს მხოლოდ ერთი აქტიური სესია.

# MS SQL Server-ის მზ-ს განახლება QTA-ს გამოყენებით

90

მონაცემთა ბაზის განახლების ახალი სესიის შექმნა

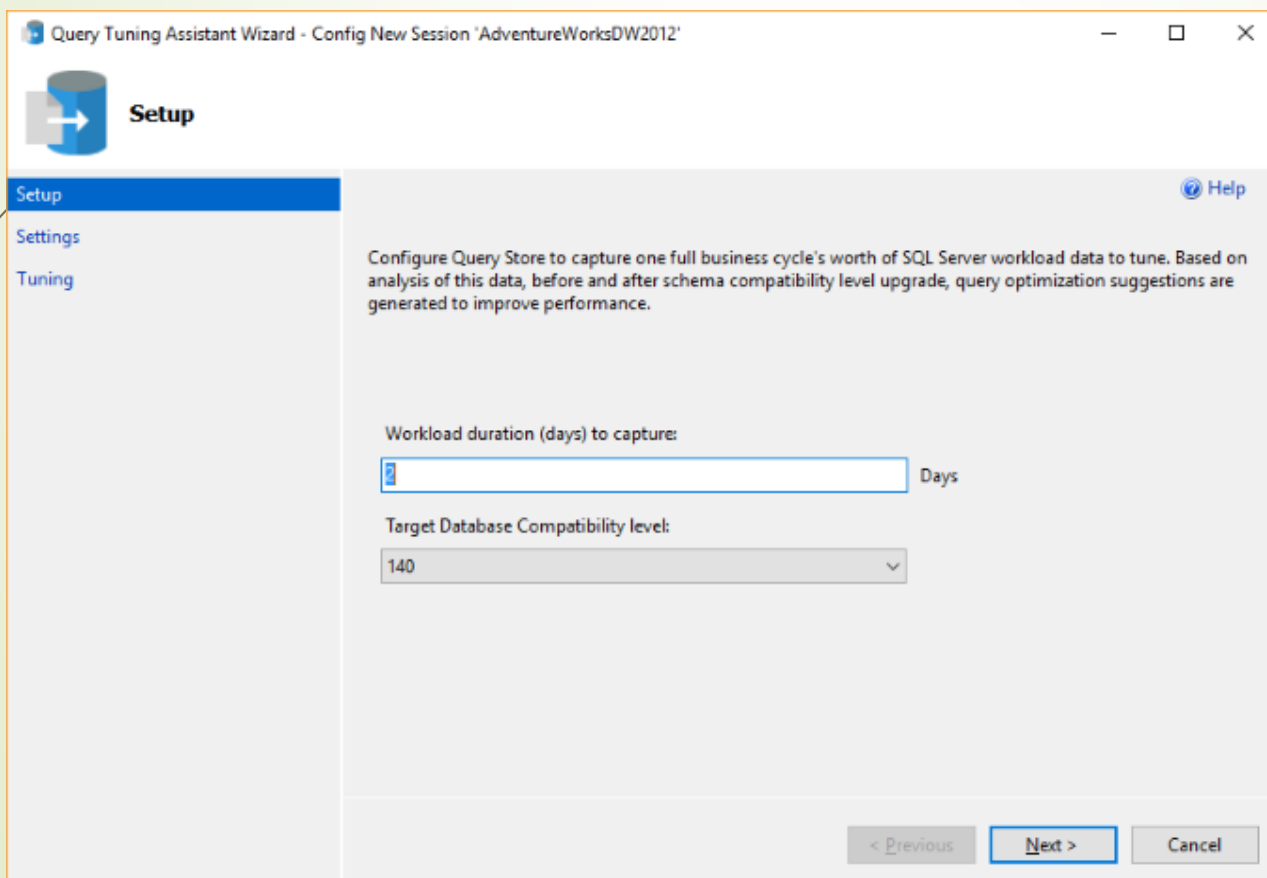


SQL Server Management Studio-ში მაუსის მარჯვენა ღილაკით უნდა გააქტიურდეს იმ მონაცემთა ბაზის სახელი, რომლისთვისაც საჭიროა თავსებადობის დონის ამაღლება. შემდეგ უნდა გააქტიურდეს სტრიქონები მიმდევრობით: „ამოცანები“ (Tasks), „მონაცემთა ბაზის განახლება“ (Database Upgrade) და „მონაცემთა ბაზის განახლების ახალი სესია“ (New Database Upgrade Session).

# MS SQL Server-ის მბ-ს განახლება QTA-ს გამოყენებით

QTA-ს ოსტატის პირველ დიალოგურ ფანჯარაში მომხმარებელმა უნდა მიუთითოს:

- მოსალოდნელი დატვირთვის ხანგრძლივობა დღეებში (მინიმუმ 1 დღე);
- მიზნობრივი მონაცემთა ბაზის თავსებადობის დონე, რომლითაც უნდა გააჩნდეს მომხმარებლის მონაცემთა ბაზას QTA სამუშაო პროცესის დასრულების შემდეგ.



შემდეგ დიალოგურ ფანჯარაზე გადასასვლელად ღილაკი „შემდეგი“ (Next).

(გაგრძელება შემდეგ სლაიდზე)



# MS SQL Server-ის მბ-ს განახლება QTA-ს გამოყენებით

შემდეგი დიალოგური ფანჯარა არის პარამეტრების ფანჯარა, რომელიც მოცემულია ორი სვეტის გამოსახულებით: მიმდინარე პარამეტრები და რეკომენდირებული პარამეტრები. რეკომენდირებული პარამეტრები ნაგულისხმევად არის შერჩეული, მაგრამ მომხმარებელს ეძლევა შესაძლებლობა გადაერთოს მიმდინარე სვეტზე და განახორციელოს შეხედულებისამებრ პარამეტრების ცვლილება.

Query Tuning Assistant Wizard - Config New Session 'AdventureWorksDW2012'

**Settings**

Setup  
Settings  
Tuning

Query Store must be configured to capture the requested data. Choose from the recommended settings below, or edit your current configuration to ensure data collection covers pre-upgrade and post-upgrade workload, based on what was set as the duration to capture:

	<input type="radio"/> Current	<input checked="" type="radio"/> Recommended	
Operation Mode	Off	Read write	
Data Flush Interval	900	900	seconds
Statistics Collection Interval	60	60	minutes
Max. Size	100	1024	MB
Query Store Capture Mode	All	Auto	
Size-Based Cleanup Mode	Auto	Auto	
Stale Query Threshold	30	4	Days

< Previous   **Next >**   Cancel

შემდეგ დიალოგურ ფანჯარაზე გადასასვლელად ღილაკი „შემდეგი“ (Next).

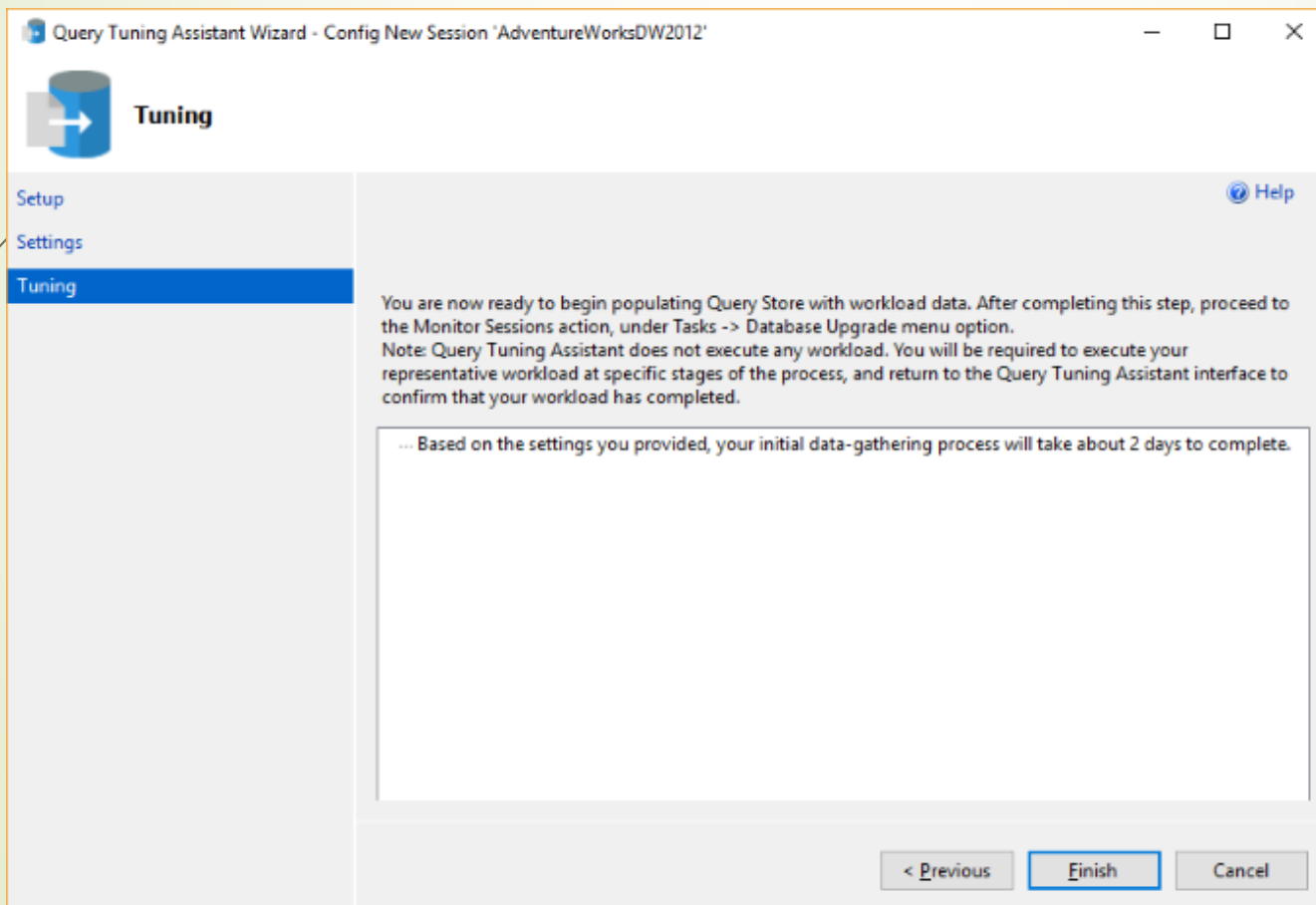
(გაგრძელება შემდეგ სლაიდზე)



# MS SQL Server-ის მბ-ს განახლება QTA-ს გამოყენებით

შემდეგ დიალოგურ ფანჯარაში ხორციელდება სეანსის კონფიგურაციას დასკვნითი მოქმედებები და შეიცავს ინფორმაციას სეანსის გახსნის შემდგომ მოქმედებებზე და მასთან მუშაობაზე.

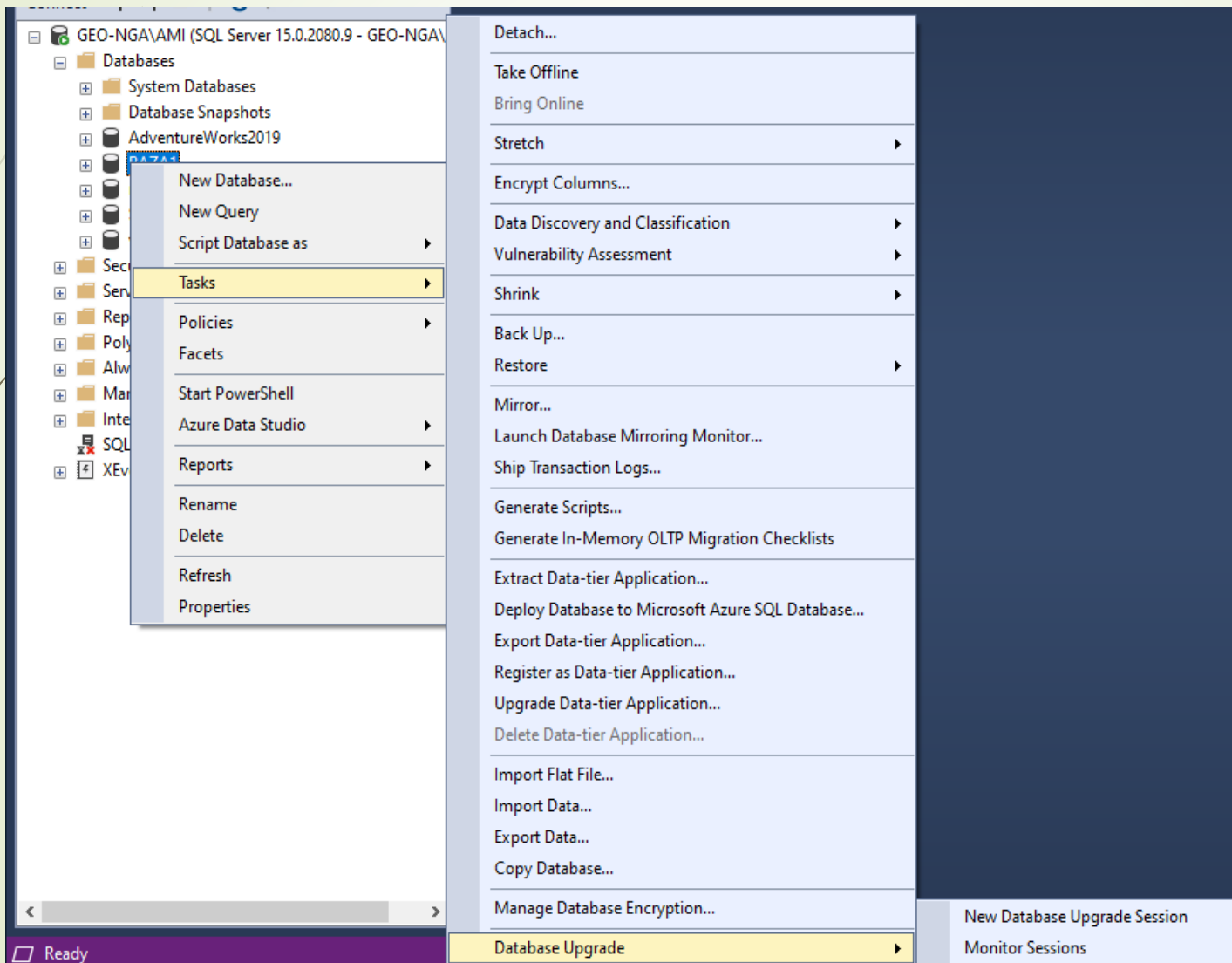
დასრულებისათვის ღილაკი „დასრულება“ (Finish).



# MS SQL Server-ის მზ-ს განახლება QTA-ს გამოყენებით

94

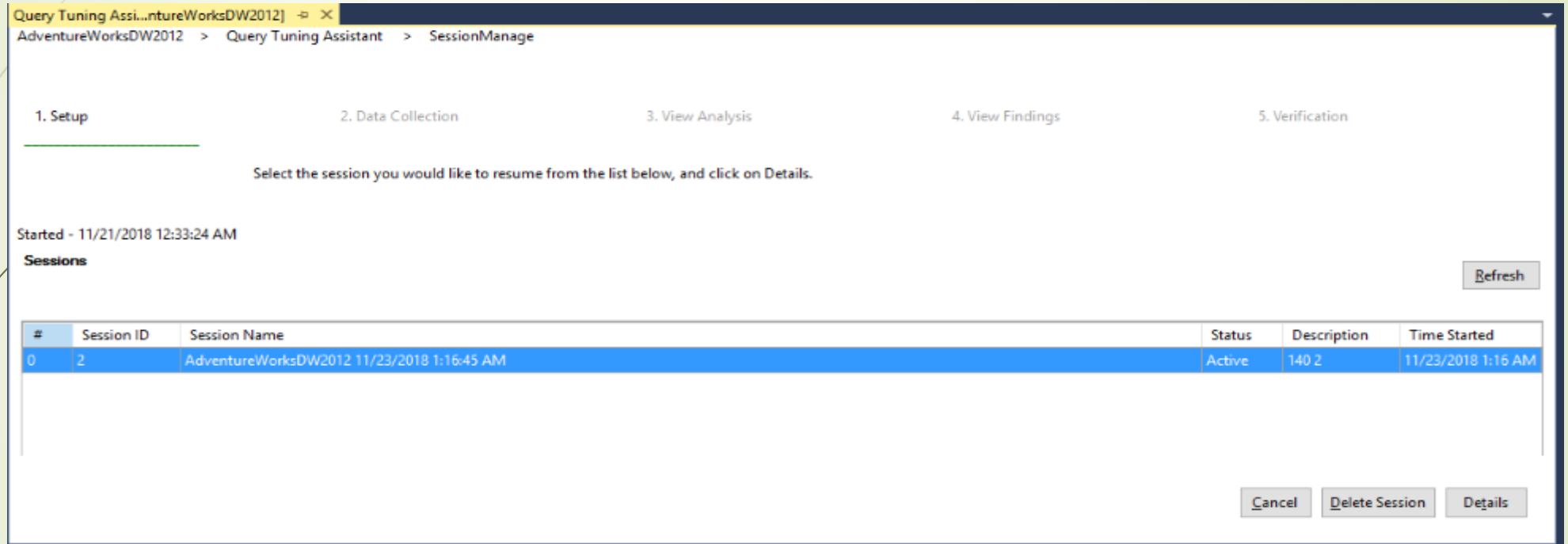
სესიების მონიტორინგი



SQL Server Management Studio-ში მაუსის მარჯვენა ღილაკით უნდა გააქტიურდეს იმ მონაცემთა ბაზის სახელი, რომლისთვისაც საჭიროა თავსებადობის დონის ამაღლება. შემდეგ უნდა გააქტიურდეს სტრიქონები მიმდევრობით: „ამოცანები“ (Tasks), „მონაცემთა ბაზის განახლება“ (Database Upgrade) და „სესიების მონიტორინგი“ (Monitor Sessions).

# MS SQL Server-ის მბ-ს განახლება QTA-ს გამოყენებით

QTA-ს მონიტორინგის პირველ დიალოგურ ფანჯარაში (1. Setup) გამოსახება მიმდინარე და წარსული სესიების სია ამ მონაცემთა ბაზისათვის, რომელშიც მომხმარებელმა უნდა შეარჩიოს საჭირო სესიის:

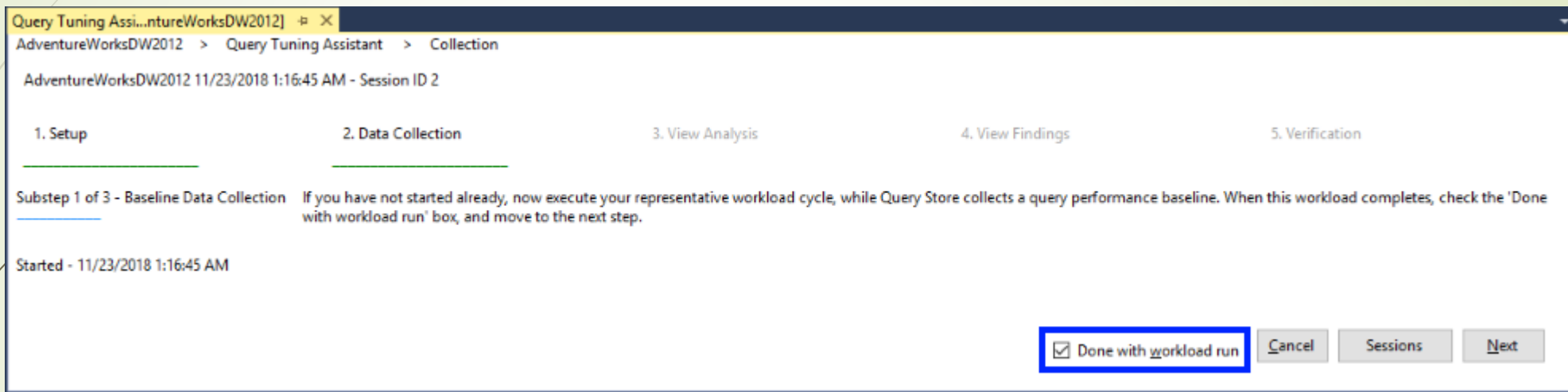


თუ მიმდინარე სესიის არ არის სიაში, მომხმარებელმა უნდა გაააქტიუროს ღილაკი „განახლება“ (Refresh), ხოლო შემდეგ დიალოგურ ფანჯარაზე გადასასვლელად ღილაკი „დეტალები“ (Details).

(გაგრძელება შემდეგ სლაიდზე)

# MS SQL Server-ის მბ-ს განახლება QTA-ს გამოყენებით

შემდეგი დიალოგური ფანჯარა საბაზისო მონაცემების კოლექცია (2. Data Collection) მომხმარებელს სთხოვს აწარმოოს წარმომადგენლობითი მუშა დატვირთვის ციკლი, რათა მონაცემთა საცავმა შეძლოს შეაგროვოს ძირითადი მონაცემები.



მუშა დატვირთვის დასრულებისათვის მომხმარებელმა უნდა მონიშნოს „დასრულება მუშა დატვირთვის შესრულებით“ (Done with workload run) და შემდეგ დიალოგურ ფანჯარაზე გადასასვლელად ღილაკი „შემდეგი“ (Next).

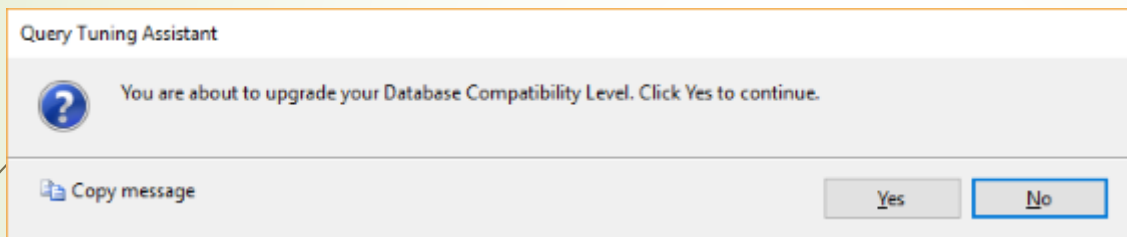
(გაგრძელება შემდეგ სლაიდზე)



# MS SQL Server-ის მბ-ს განახლება QTA-ს გამოყენებით

QTA-ს დიალოგური ფანჯარა შეიძლება დაიხუროს მუშაობის დატვირთვისას. სესიაზე დაბრუნება, რომელიც შემდგომში რჩება აქტიურ მდგომარეობაში, განახლდება იმავე საფეხურიდან, სადაც ის შეჩერებული იყო.

შემდეგ დიალოგურ ფანჯარაში სისტემა მოითხოვს მონაცემთა ბაზის თავსებადობის დონის სასურველ მიზნამდე განახლების ნებართვას.

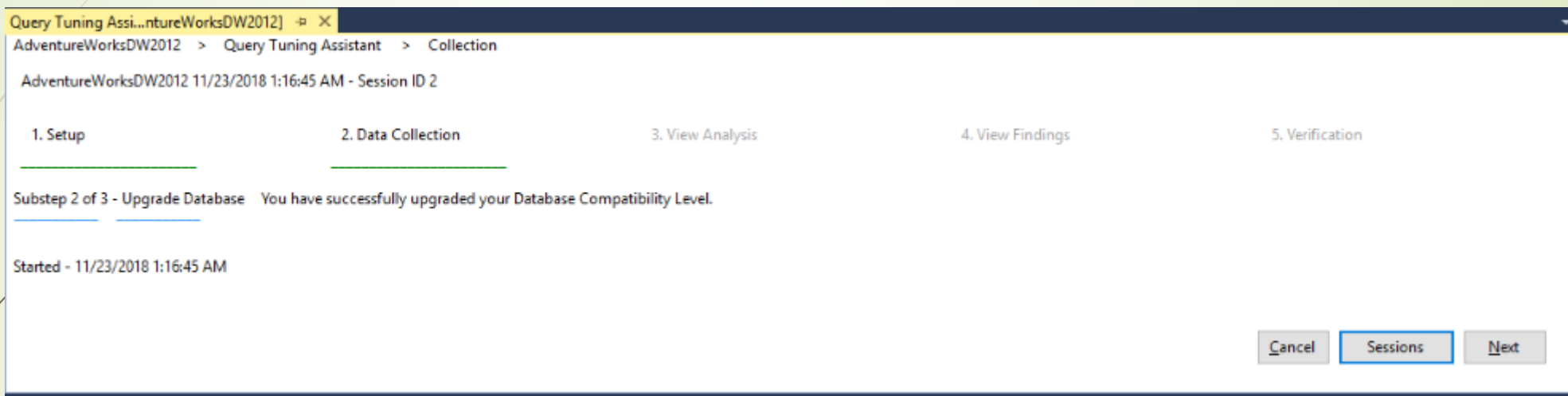


შემდეგ საფეხურზე გადასასვლელად მომხმარებელმა უნდა გაააქტიუროს ღილაკი „დიახ“ (Yes).

(გაგრძელება შემდეგ სლაიდზე)

# MS SQL Server-ის მბ-ს განახლება QTA-ს გამოყენებით

შემდეგ დიალოგურ ფანჯარაში დასტურდება, რომ მონაცემთა ბაზის თავსებადობის დონე წარმატებით იქნა განახლებული.



შემდეგ დიალოგურ ფანჯარაზე გადასასვლელად მომხმარებელმა უნდა გაააქტიუროს ლილავი „სესიები“ (Sessions).

(გაგრძელება შემდეგ სლაიდზე)

# MS SQL Server-ის მბ-ს განახლება QTA-ს გამოყენებით

99

შემდეგ დიალოგურ ფანჯარაში მომხმარებელს სთხოვს ხელახლა აწარმოოს წარმომადგენლობითი მუშა დატვირთვის ციკლი.

Query Tuning Assistant - AdventureWorksDW2012

AdventureWorksDW2012 > Query Tuning Assistant > Collection

AdventureWorksDW2012 11/23/2018 1:16:45 AM - Session ID 2

1. Setup 2. Data Collection 3. View Analysis 4. View Findings 5. Verification

Substep 3 of 3 - Observed Data Collection Now that you have upgraded to the desired Database Compatibility Level, re-run the same workload as you did before. This enables Query Store to collect a comparative baseline that will be used to search for optimization opportunities. When this workload completes, check the 'Done with workload run' box, and move to the next step.

Started - 11/23/2018 1:16:45 AM

Top 10 regressed queries

Refresh Queries to show: 10 Metric: Duration Aggregation: Avg

#	Query ID	Query Text	Runs	Baseline Metric	Observed Metric	% Change	Tunable
0	1	IF @maxWeight <= (SELECT Weight from DimProduct WHERE ProductKey = @productKey)	300	0.4	0.65	-62.50	False
1	12	SELECT OrderDateKey, SUM(SalesAmount) AS TotalSales FROM FactInternetSales GROUP BY OrderDateKey...	300	26.88	28.2	-4.91	True
2	8	SELECT NumberCarsOwned FROM DimCustomer GROUP BY YearlyIncome, NumberCarsOwned	300	10.26	10.75	-4.78	True
3	14	SELECT * FROM FactInternetSales fis INNER JOIN DimProduct dp ON fis.ProductKey = dp.ProductKey WHER...	300	1184.14	1236.23	-4.40	True
4	6	SELECT CustomerKey, SUM(SalesAmount) AS sas FROM FactInternetSales GROUP BY CustomerKey WITH (...)	300	55.38	56.66	-2.31	True
5	11	SELECT SalesAmount FROM FactInternetSales GROUP BY SalesAmount, SalesAmount*1.10	300	50.61	51.41	-1.58	True
6	10	SELECT SalesAmount, SalesAmount*1.10 SalesTax FROM FactInternetSales GROUP BY SalesAmount	300	23.05	23.35	-1.30	True
7	9	SELECT (SalesAmount + TaxAmt + Freight) AS TotalCost FROM FactInternetSales GROUP BY SalesAmount, T...	300	33.22	33.29	-0.21	True
8	2	(SELECT @productKey AS ProductKey, EnglishDescription, Weight, 'This product is too heavy to ship and i...	300	0.02	0.02	0.00	False
9	13	SELECT TOP (10) r.ResellerName, r.AnnualSales FROM DimReseller AS r ORDER BY AnnualSales DESC, Resell...	300	0.56	0.56	0.00	True

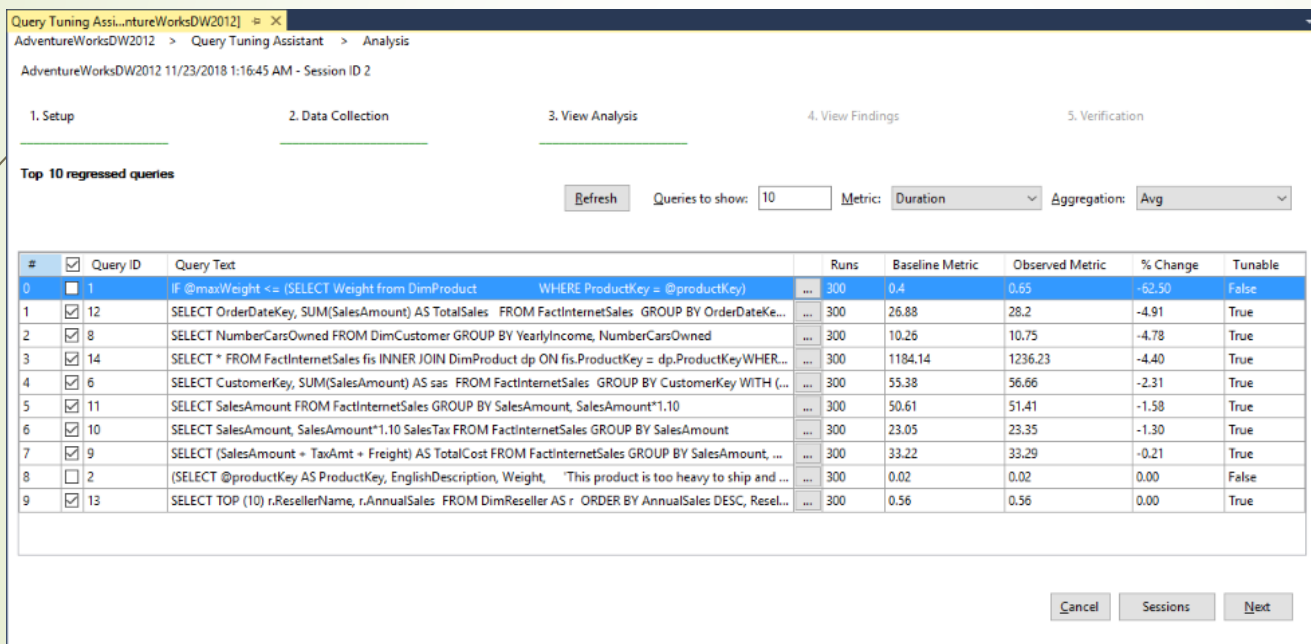
☒ Done with workload run Cancel Sessions Next

მუშა დატვირთვის დასრულებისათვის მომხმარებელმა უნდა მონიშნოს „დასრულება მუშა დატვირთვის შესრულებით“ (Done with workload run) და შემდეგ დიალოგურ ფანჯარაზე გადასასვლელად ღილაკი „შემდეგი“ (Next).

# MS SQL Server-ის მბ-ს განახლება QTA-ს გამოყენებით

100

შემდეგ დიალოგურ ფანჯარაში „ანალიზის ნახვა“ (3. View Analysis) მომხმარებელს ეძლევა შესაძლებლობა შეარჩიოს მოთხოვნები ექსპერიმენტისთვის და იპოვოს ოპტიმიზაციის შესაძლებლობები. მოთხოვნების შერჩევაზე დაბრუნება ამ დიალოგური ფანჯრიდან გადასვლის შემდეგ შეუძლებელია და ამიტომ, თუ არ იქნა შერჩეული ყველა სასურველი მოთხოვნა, მომხმარებელმა მოგვიანებით უნდა შექმნას ახალი სესია და გაიმეოროს ყველა მოქმედებები.



Query Tuning Assistant [AdventureWorksDW2012] > Query Tuning Assistant > Analysis

AdventureWorksDW2012 11/23/2018 1:16:45 AM - Session ID 2

1. Setup 2. Data Collection 3. View Analysis 4. View Findings 5. Verification

Top 10 regressed queries

Refresh Queries to show: 10 Metric: Duration Aggregation: Avg

#	Query ID	Query Text	Runs	Baseline Metric	Observed Metric	% Change	Tunable
0	1	IF @maxWeight <= (SELECT Weight from DimProduct WHERE ProductKey = @productKey)	300	0.4	0.65	-62.50	False
1	12	SELECT OrderDateKey, SUM(SalesAmount) AS TotalSales FROM FactInternetSales GROUP BY OrderDateKey...	300	26.88	28.2	-4.91	True
2	8	SELECT NumberCarsOwned FROM DimCustomer GROUP BY YearlyIncome, NumberCarsOwned	300	10.26	10.75	-4.78	True
3	14	SELECT * FROM FactInternetSales fis INNER JOIN DimProduct dp ON fis.ProductKey = dp.ProductKey WHERE...	300	1184.14	1236.23	-4.40	True
4	6	SELECT CustomerKey, SUM(SalesAmount) AS sas FROM FactInternetSales GROUP BY CustomerKey WITH (...)	300	55.38	56.66	-2.31	True
5	11	SELECT SalesAmount FROM FactInternetSales GROUP BY SalesAmount, SalesAmount*1.10	300	50.61	51.41	-1.58	True
6	10	SELECT SalesAmount, SalesAmount*1.10 SalesTax FROM FactInternetSales GROUP BY SalesAmount	300	23.05	23.35	-1.30	True
7	9	SELECT (SalesAmount + TaxAmt + Freight) AS TotalCost FROM FactInternetSales GROUP BY SalesAmount, ...	300	33.22	33.29	-0.21	True
8	2	(SELECT @productKey AS ProductKey, EnglishDescription, Weight, 'This product is too heavy to ship and ...	300	0.02	0.02	0.00	False
9	13	SELECT TOP (10) r.ResellerName, r.AnnualSales FROM DimReseller AS r ORDER BY AnnualSales DESC, Resel...	300	0.56	0.56	0.00	True

Cancel Sessions Next

ექსპერიმენტის დასაწყებად ღილაკი „შემდეგი“ (Next).  
(გაგრძელება შემდეგ სლაიდზე)



# MS SQL Server-ის მბ-ს განახლება QTA-ს გამოყენებით

შემდეგ დიალოგურ ფანჯარაში „შედეგების ჩვენება“ (View Findings) მომხმარებელს ეძლევა შესაძლებლობა შეარჩიოს გეგმის სტრუქტურისათვის თუ რომელი მოთხოვნები გამოიყენოს შემოთავაზებული გადაწყვეტილებისათვის.

Query Tuning Assistant - AdventureWorksDW2012

AdventureWorksDW2012 > Query Tuning Assistant > Findings

AdventureWorksDW2012 11/23/2018 1:16:45 AM - Session ID 2

1. Setup      2. Data Collection      3. View Analysis      4. View Findings      5. Verification

#	<input type="checkbox"/> Query ID	Query Text	Status	Baseline Metric	Observed Metric	% Change	Query Option	Can Deploy
0	<input checked="" type="checkbox"/> 12	SELECT OrderDateKey, SUM(SalesAmount) AS T...	Test complete	26.88	17	36.76	<a href="https://go.microsoft.com/fwlink/?linkid=2028175">https://go.microsoft.com/fwlink/?linkid=2028175</a>	True
1	<input type="checkbox"/> 8	SELECT NumberCarsOwned FROM DimCustom...	Test complete	10.26	7	31.77	<a href="https://go.microsoft.com/fwlink/?linkid=2028175">https://go.microsoft.com/fwlink/?linkid=2028175</a>	True
2	<input type="checkbox"/> 14	SELECT * FROM FactInternetSales fis INNER JOI...	Test complete	1184.14	484	59.13	<a href="https://go.microsoft.com/fwlink/?linkid=2028217">https://go.microsoft.com/fwlink/?linkid=2028217</a>	True
3	<input type="checkbox"/> 6	SELECT CustomerKey, SUM(SalesAmount) AS s...	Test complete	55.38	32	42.22	<a href="https://go.microsoft.com/fwlink/?linkid=2028175">https://go.microsoft.com/fwlink/?linkid=2028175</a>	True
4	<input type="checkbox"/> 11	SELECT SalesAmount FROM FactInternetSales G...	Test complete	50.61	32	36.77	<a href="https://go.microsoft.com/fwlink/?linkid=2028175">https://go.microsoft.com/fwlink/?linkid=2028175</a>	True
5	<input type="checkbox"/> 10	SELECT SalesAmount, SalesAmount*1.10 SalesT...	Test complete	23.05	14	39.26	<a href="https://go.microsoft.com/fwlink/?linkid=2028175">https://go.microsoft.com/fwlink/?linkid=2028175</a>	True
6	<input type="checkbox"/> 9	SELECT (SalesAmount + TaxAmt + Freight) AS ...	Test complete	33.22	21	36.79	<a href="https://go.microsoft.com/fwlink/?linkid=2028175">https://go.microsoft.com/fwlink/?linkid=2028175</a>	True
7	<input type="checkbox"/> 13	SELECT TOP (10) r.ResellerName, r.Annu...	Test complete	0.56	0	100.00	<a href="https://go.microsoft.com/fwlink/?linkid=2028175">https://go.microsoft.com/fwlink/?linkid=2028175</a>	True

Hints force certain behaviors that may get addressed in subsequent updates. Microsoft recommends you only apply hints when no other option exists, and that you plan to revisit hinted code with every new upgrade. By forcing behaviors, you may not allow your workload to benefit from enhancements introduced in newer versions.

Cancel   Sessions   Deploy

შემდეგ დიალოგურ ფანჯარაზე გადასასვლელად მომხმარებელმა უნდა გაააქტიუროს ლილაკი „სესიები“ (Sessions).

(გაგრძელება შემდეგ სლაიდზე)

# MS SQL Server-ის მბ-ს განახლება QTA-ს გამოყენებით

შემდეგი დიალოგური ფანჯარა „შემოწმება“ (Verification) წინა ფანჯრისგან განსხვავდება მხოლოდ ბოლო სვეტით: წინაში იყო „შესაძლებელია გაშლა“ (Can Deploy), ხოლო ამაში – „შესაძლებელია უკუქცევა“ (Can Rollback), რომლის მითითებისას სტრიქონი უკუიქცევა.

Query Tuning Assistant - AdventureWorksDW2012

AdventureWorksDW2012 > Query Tuning Assistant > Verification

AdventureWorksDW2012 11/23/2018 1:16:45 AM - Session ID 2

1. Setup      2. Data Collection      3. View Analysis      4. View Findings      5. Verification

You can choose one or more queries to rollback any previously applied recommendations. For example, when upgrading to a newer version of the Database Engine, you should plan to rollback previous recommendations that may no longer be applicable.

Started - 11/23/2018 1:16:45 AM

#	<input type="checkbox"/> Query ID	Query Text	Status	Baseline Metric	Observed Metric	% Change	Query Option	Can Rollback
0	<input checked="" type="checkbox"/> 12	SELECT OrderDateKey, SUM(SalesAmount) AS Tot...	Deployed	26.88	17	36.76	<a href="https://go.microsoft.com/fwlink/?linkid=2028175">https://go.microsoft.com/fwlink/?linkid=2028175</a>	True
1	<input type="checkbox"/> 14	SELECT * FROM FactInternetSales fis INNER JOIN ...	Deployed	1184.14	484	59.13	<a href="https://go.microsoft.com/fwlink/?linkid=2028217">https://go.microsoft.com/fwlink/?linkid=2028217</a>	True
2	<input type="checkbox"/> 6	SELECT CustomerKey, SUM(SalesAmount) AS sas ...	Deployed	55.38	32	42.22	<a href="https://go.microsoft.com/fwlink/?linkid=2028175">https://go.microsoft.com/fwlink/?linkid=2028175</a>	True
3	<input type="checkbox"/> 11	SELECT SalesAmount FROM FactInternetSales GR...	Deployed	50.61	32	36.77	<a href="https://go.microsoft.com/fwlink/?linkid=2028175">https://go.microsoft.com/fwlink/?linkid=2028175</a>	True
4	<input type="checkbox"/> 10	SELECT SalesAmount, SalesAmount*1.10 SalesTax...	Deployed	23.05	14	39.26	<a href="https://go.microsoft.com/fwlink/?linkid=2028175">https://go.microsoft.com/fwlink/?linkid=2028175</a>	True
5	<input type="checkbox"/> 9	SELECT (SalesAmount + TaxAmt + Freight) AS To...	Deployed	33.22	21	36.79	<a href="https://go.microsoft.com/fwlink/?linkid=2028175">https://go.microsoft.com/fwlink/?linkid=2028175</a>	True
6	<input type="checkbox"/> 13	SELECT TOP (10) r.ResellerName, r.Annu...	Deployed	0.56	0	100.00	<a href="https://go.microsoft.com/fwlink/?linkid=2028175">https://go.microsoft.com/fwlink/?linkid=2028175</a>	True

Cancel   Sessions   Rollback

პროცესის დასრულებისათვის და განახლების დაწყებისათვის მომხმარებელმა უნდა გაააქტიუროს ღილაკი „სესიები“ (Sessions).

# 4 LEQC

# დიდი მოცულობის კოპირების პროგრამა **bulk copy (bcp)**

bcp გამოიყენება მონაცემების მასიური კოპირებისათვის Microsoft SQL Server-სა და მომხმარებლის მიერ განსაზღვრულ ფორმატში მონაცემთა ფაილს შორის. bcp-ს გამოყენებით შესაძლებელია SQL Server-ის ცხრილებში ახალი სტრიქონების დიდი რაოდენობით იმპორტი ან SQL Server-ის ცხრილებიდან მონაცემთა ფაილებში მონაცემების ექსპორტი. queryout მოთხოვნის პარამეტრის გამოყენებისას გარდა, პროგრამა არ საჭიროებს Transact-SQL ენის ცოდნას.

SQL Server-ის ცხრილში მონაცემების იმპორტირებისთვის უნდა იქნეს გამოყენებული ფორმატის ფაილი, რომელიც შეიქმნა ამ ცხრილისთვის ან უნდა ვიცოდეთ ცხრილის სტრუქტურა და მისი სვეტებისთვის დაშვებული მონაცემთა ტიპები.

bcp-ს მონაცემთა სარეზერვო ასლის შექმნისათვის გამოყენებისას, უნდა შეიქმნას ფორმატის ფაილი მონაცემთა ფორმატის დასაწერად, რადგან bcp მონაცემთა ფაილები არ შეიცავენ ინფორმაციას სქემის ან ფორმატის შესახებ და თუ ცხრილი ან წარმოდგენა წაიშლება ფორმატის ფაილის უქონლობის შემთხვევაში მონაცემების იმპორტი შეუძლებელია.

bcp-ს ბოლო ვერსია შესაძლებელია ჩამოტვირთულ იქნეს შემდეგი ბმულიდან:

**[Download Microsoft Command Line Utilities 15 for SQL Server \(x64\)](#)**



BCP-ს ინსტალირების წინ უნდა შემოწმდეს თავსებადობის და არსებობის პირობები:

1. სისტემისადმი მოთხოვნები: **Windows 10, Windows 7, Windows 8, Windows 8.1, Windows Server 2008, Windows Server 2008 R2, Windows Server 2008 R2 SP1, Windows Server 2012, Windows Server 2012 R2, Windows Server 2016, Windows Server 2019**
2. სისტემაში უნდა არსებობდეს: [Windows Installer 4.5](#) და [Microsoft ODBC Driver 17 for SQL Server](#).
3. BCP ვერსიის შესამოწმებლად უნდა შესრულდეს ბრძანება **bcp /v** (დაადასტურეთ, რომ გამოიყენება 15.0.2000.5 ან უფრო მაღალი ვერსია).

მხოლოდ ამ პირობების შემოწმების შემდეგ არის bcp-ს გამოყენების შესაძლებლობა, რომლის შესრულებისათვის შესაბამისი ბრძანება უნდა შევიტანოთ Command Prompt-ის ფანჯარაში (Start → All Programs → Accessories → Command Prompt).

BCP-ს შედეგები არ აისახება ტრანზაქციის ჟურნალში, ანუ თუ გადაწერის ოპერაცია წარუმატებლად დამთავრდა, მაშინ საწყისი მონაცემები ვერ აღდგება. BCP მუშაობს ავტომატურ (თუ დაფორმატების გასაღებები მითითებულია ბრძანების სტრიქონში) ან ინტერაქტიურ რეჟიმში (დაფორმატების გასაღებების გარეშე და მაშინ გაიცემა შესაბამისი შეკითხვები).

BCP-ს გააჩნია შემდეგი სინტაქსი:

**bcp [database\_name.] schema.{table\_name | view\_name | "query"}  
 {in data\_file | out data\_file | queryout data\_file | format nul}**

<b>[-a packet_size]</b>	<b>[-m max_errors]</b>
<b>[-b batch_size]</b>	<b>[-n]</b>
<b>[-c]</b>	<b>[-N]</b>
<b>[-C { ACP   OEM   RAW   code_page } ]</b>	<b>[-o output_file]</b>
<b>[-d database_name]</b>	<b>[-P password]</b>
<b>[-D]</b>	<b>[-q]</b>
<b>[-e err_file]</b>	<b>[-r row_term]</b>
<b>[-E]</b>	<b>[-R]</b>
<b>[-f format_file]</b>	<b>[-S [server_name\instance_name]]</b>
<b>[-F first_row]</b>	<b>[-t field_term]</b>
<b>[-G Azure Active Directory Authentication]</b>	<b>[-T]</b>
<b>[-h "hint [...n]" ]</b>	<b>[-U login_id]</b>
<b>[-i input_file]</b>	<b>[-v]</b>
<b>[-k]</b>	<b>[-V (80   90   100   110   120   130 ) ]</b>
<b>[-K application_intent]</b>	<b>[-w]</b>
<b>[-l login_timeout]</b>	<b>[-x]</b>
<b>[-L last_row]</b>	

განვიხილოთ BCP-ს არგუმენტები:

**data\_file** - არის მონაცემთა ფაილის სრული გზა. როდესაც ხორციელდება მონაცემების იმპორტირება SQL Server-ში, მონაცემთა ფაილი შეიცავს მონაცემებს, რომლებიც კოპირდება მითითებულ ცხრილში ან წარმოდგენაში. როდესაც SQL Server-დან ხდება მონაცემთა დიდი მოცულობის ექსპორტი, მონაცემთა ფაილი შეიცავს ცხრილიდან ან წარმოდგენიდან კოპირებულ მონაცემებს. გზას შეიძლება ჰქონდეს 1 – დან 255 სიმბოლომდე. მონაცემთა ფაილი შეიძლება შეიცავდეს მაქსიმუმ  $2^{63} - 1$  მწკრივს.

**database\_name** - მონაცემთა ბაზის სახელია, რომელშიც მდებარეობს მითითებული ცხრილი ან წარმოდგენა. თუ იგი არ არის მითითებული, მაშინ განიხილება მომხმარებლის ნაგულისხმევი მონაცემთა ბაზა. ასევე შესაძლებელია პირდაპირ მიეთითოს მონაცემთა ბაზის სახელი -d-ს გამოყენებით.

**in data\_file | out data\_file | queryout data\_file | format nul** - განსაზღვრავს ასლის მიმართულებას, შემდეგნაირად:

➤ **in** - ასლებიდან ფაილიდან მონაცემთა ბაზის ცხრილში ან წარმოდგენაში;



- **out** - გადაწეს მონაცემებს მონაცემთა ბაზის ცხრილიდან ან წარმოდგენიდან მონაცემთა ფაილში. თუ მიუთითებთ არსებულ ფაილს, ფაილი გადაიწერა. მონაცემთა ამოღებისას bcp პროგრამა წარმოადგენს ცარიელ სტრიქონს როგორც null-ს და null სტრიქონს როგორც ცარიელ სტრიქონს;
- **queryout** – აკოპირებს მონაცემებს მოთხოვნიდან (გამოიყენება მხოლოდ მოთხოვნიდან მონაცემების მასობრივი კოპირებისას);
- **format** - ქმნის ფორმატის ფაილს დამყარებულს მითითებულ პარამეტრებზე (-n, -c, -w, ან -N) და ცხრილის ან წარმოდგენის გამყოფებზე. მონაცემთა მასიურ კოპირებისას bcp პროგრამამ შეიძლება მიმართოს ფორმატის ფაილს, რაც ზოგავს დროს და საშუალებებს ფორმატის ინფორმაციის ინტერაქტიულად განმეორებით შეტანისას. Format პარამეტრი მოითხოვს -f პარამეტრს, ხოლო XML-ფაილის შექმნისას ასევე მოითხოვს -x პარამეტრს. მნიშვნელობად უნდა მიუთითოთ nul (format nul).

**owner** მფლობელი - არის ცხრილის ან წარმოდგენის მფლობელის სახელი. თუ ოპერაციის განმახორციელებელი მომხმარებელი არის მითითებული ცხრილის ან წარმოდგენის მფლობელი, პარამეტრი შეიძლება არ მიეთითოს, სხვა შემთხვევაში SQL Server აბრუნებს შეცდომის შეტყობინებას და ოპერაცია გაუქმდება.



"**query**" არის Transact-SQL მოთხოვნა, რომელიც აბრუნებს შედეგების ნაკრებებს. თუ მოთხოვნა აბრუნებს მრავალ შედეგების ნაკრებს, მონაცემების ფაილში კოპირდება მხოლოდ პირველი შედეგების ნაკრები (შემდეგი შედეგების ნაკრებები იგნორირდება). მოთხოვნაში გამოიყენება ორმაგი ბრჭყალები, ხოლო მოთხოვნაში ჩასმული გამოსახულებებისათვის გამოიყენება ბრჭყალები. მოთხოვნიდან მონაცემთა მასიური კოპირებისას უნდა იქნეს მითითებული ასევე არგუმენტი queryout.

მოთხოვნას შეუძლია მიმართოს შენახულ პროცედურას მხოლოდ იმ შემთხვევაში, თუ შენახულ პროცედურაში მითითებული ყველა ცხრილი აშკარად არსებობს, მანამ სანამ bcp გაეშვება.

**table\_name** ცხრილის\_სახელი - არის დანიშნულების ცხრილის სახელი SQL Server-ში მონაცემების იმპორტირებისას (**in**), და წყარო ცხრილის SQL Server-დან მონაცემების ექსპორტისას (**out**).

**view\_name** წარმოდგენის\_სახელი - არის SQL Server (**in**) მონაცემების კოპირებისას დანიშნულების წარმოდგენის სახელი და SQL Server (**out**) მონაცემების კოპირებისას წყარო-წარმოდგენის. მიზნობრივი წარმოდგენად შეიძლება გამოყენებულ იქნეს მხოლოდ წარმოდგენები, რომლების ყველა სვეტი ერთსა და იმავე ცხრილს ემყარება.

**-a packet\_size** - პაკეტის\_ზომა განსაზღვრავს Server-დან Server-ზე გაგზავნილ ქსელის თითო პაკეტში ბაიტთა რაოდენობას. SQL Server-ის კონფიგურაციის პარამეტრის დაყენება შესაძლებელია SQL Server Management Studio (ან sp\_configure სისტემის შენახული პროცედურის) გამოყენებით. packet\_size შეიძლება იყოს 4096-დან 65535 ბაიტამდე (ნაგულისხმევია 4096). პაკეტის გაზრდილმა ზომამ შეიძლება დაასწრაფოს ოპერაციების შესრულება, ხოლო თუ უფრო დიდი პაკეტია მოთხოვნილი, რომლის გაცემა შეუძლებელია, გამოყენებული იქნება ნაგულისხმევი მნიშვნელობა.

**-b batch\_size** - პარტიის ზომა მიუთითებს იმპორტირებული მონაცემებში ერთ პაკეტში მწკრივების რაოდენობას. თითოეული პარტია იმპორტირდება და რეგისტრირდება როგორც ცალკე ოპერაცია, რომელიც ფიქსირდება სრული პაკეტის იმპორტირების დასრულებისას. შეთანხმებით მონაცემთა ფაილიდან ყველა სტრიქონის იმპორტირება ხორციელდება ერთ პარტიაში. მწკრივების მრავალ პარტიის შორის განაწილებისათვის უნდა მიეთითოს batch\_size, რომელიც ნაკლებია მონაცემთა ფაილში მწკრივების რაოდენობაზე. თუ რაიმე პარტიის ტრანზაქცია ჩაიშალა, მხოლოდ მიმდინარე პარტიის ჩანართები ბრუნდება უკან. წარმოქმნილი შეცდომა არ შეეხება უკვე დაფიქსირებულ ტრანზაქციებს (არ შეიძლება ამ პარამეტრის გამოყენება -h პარამეტრთან ერთად "ROWS\_PER\_BATCH = bb").



**-c** - ასრულებს ოპერაციას სიმბოლური (ტექსტური) მონაცემთა ტიპის გამოყენებით. ეს პარამეტრი არ ითხოვს მონაცემთა ტიპის მითითებას თითოეულ სვეტისათვის. მონაცემთა შენახვისათვის გაიყენებს ტიპს char, პრეფიქსების გარეშე და \t (ტაბულაციის სიმბოლო), ველების გამოყოფად, ხოლო სტრიქონის დასრულების და ახალ სტრიქონზე გადასვლისათვის გამოიყენება \r \n. **-c** არ არის თავსებადი **-w**-სთან.

**-C { ACP | OEM | RAW | code\_page }** - განსაზღვრავს მონაცემთა კოდის გვერდს მონაცემთა ფაილში. code\_page მნიშვნელოვანია მხოლოდ იმ შემთხვევაში, თუ მონაცემები შეიცავს char, varchar ან text სვეტებს სიმბოლოების მნიშვნელობებით 127-ზე მეტი ან 32-ზე ნაკლები. რეკომენდირებულია ფორმატირების ფაილში მითითებულ იქნეს სორტირების პარამეტრები ყველა სვეტისათვის, გარდა იმ შემთხვევისა, როდესაც სასურველია 65001 პარამეტრის უპირატესობა სორტირების ან კოდის გვერდის სპეციფიკაციასთან.

**-d \*\*მონაცემთა ბაზის\_სახელი\*\*** - განსაზღვრავს მონაცემთა ბაზას, რომელსაც უნდა მიუერთდეს. შეთანხმებით bcp.exe უერთდება მომხმარებლის მონაცემთა ბაზას. თუ მითითებულია -d მონაცემთა ბაზის\_სახელი და სამი ნაწილის სახელი (database\_name.schema.table როგორც bcp.exe- სპირველი პარამეტრი), მოხდება შეცდომა, რადგან მონაცემთა ბაზის სახელს ორჯერ ვერ მიუთითებთ. თუ მონაცემთა ბაზის\_სახელი იწყება დეფისით (-) ან გადაკვეთის ხაზით (/), არ უნდა დაემატოს ადგილი - d და მონაცემთა ბაზის სახელს შორის.

**-D** - იწვევს bcp-ს **-S** პარამეტრისთვის გადაცემული მნიშვნელობის ინტერპრეტაციას, როგორც მონაცემთა წყაროს სახელს (DSN).

**-e err\_file** - განსაზღვრავს შეცდომის ფაილის სრულ გზას, რომელიც გამოიყენება იმ მწკრივების შესანახად, რომელსაც bcp ვერ გადასცემს ფაილიდან მონაცემთა ბაზაში. შეცდომის შეტყობინებები bcp ბრძანებიდან გადადის მომხმარებლის სამუშაო სადგურზე. თუ ეს პარამეტრი არ არის გამოყენებული, შეცდომის ფაილი არ შეიქმნება. თუ err\_file იწყება დეფისით (-) ან წინა გადაკვეთით (/), არ უნდა იქნეს გამოყენებული სივრცე -e- ს და err\_file მნიშვნელობას შორის.

**-E** - მიუთითებს, რომ იდენტიფიკატორების მნიშვნელობა ან მნიშვნელობები იმპორტირებულ მონაცემთა ფაილში გამოყენებული იქნება იდენტიფიკატორების სვეტისთვის. თუ -E არგუმენტი არ არის მითითებული, იმპორტირებული მონაცემების ფაილში ამ სვეტის იდენტიფიკატორის მნიშვნელობები იგნორირებულია და SQL Server-ი ავტომატურად ანიჭებს უნიკალურ მნიშვნელობებს ცხრილის შექმნისას მითითებულ საწყის და ნაზარდის მნიშვნელობებზე დაყრდნობით. დამატებითი ინფორმაციისთვის იხილეთ DBCC CHECKIDENT. თუ მონაცემთა ფაილი არ შეიცავს იდენტიფიკატორის სვეტის მნიშვნელობებს ცხრილში ან წარმოდგენაში უნდა იქნეს გამოყენებული ფორმატის ფაილი, რომელშიც მიეთითება, რომ მონაცემების იმპორტისას გამოტოვებულია ცხრილის ან წარმოდგენის იდენტიფიკატორის სვეტი. SQL Server-ი ავტომატურად მიანიჭებს უნიკალურ მნიშვნელობებს სვეტს. -E პარამეტრის გამოსაყენებლად საჭიროა სპეციალური ნებართვები.



**-f format\_file** - განსაზღვრავს ფორმატის ფაილის სრულ გზას. ამ პარამეტრის მნიშვნელობა დამოკიდებულია გარემოზე, რომელშიც ის გამოიყენება, შემდეგი წესების დაცვით:

- ▶ თუ **-f** პარამეტრი გამოიყენება **format** პარამეტრთან ერთად, მითითებული ცხრილისთვის ან წარმოდგენისთვის იქმნება ფაილი format\_file. XML ფორმატის ფაილის შესაქმნელად, უნდა მიეთითოს **-x** პარამეტრი.
- ▶ თუ გამოიყენება **in** ან **out** პარამეტრი, **-f**-სთვის საჭიროა არსებობდეს ფორმატის ფაილი (არ არის ფორმატის ფაილის გამოყენების აუცილებლობა **in** ან **out** პარამეტრით. თუ არ არის მითითებული **-f**, **-n**, **-c**, **-w** ან **-N** პარამეტრი სისტემა მოითხოვს ფორმატირების ინფორმაციას და იძლევა პასუხების დამახსოვრების შესაძლებლობას ფორმატის ფაილში (ფაილის ნაგულისხმევი სახელია bcp.fmt).
- ▶ თუ format\_file იწყება დეფისით (-) ან გადაკვეთის ხაზი (/), მაშინ **-f** და format\_file მნიშვნელობებს შორის არ უნდა იქნეს სივრცე.

**-F first\_row** - განსაზღვრავს პირველი სტრიქონის ნომერს (ექსპორტის და იმპორტის ორივე შემთხვევაში). ეს პარამეტრი უნდა აღემატებოდეს (>) 0-ს, მაგრამ ნაკლებია (<) ან ტოლია (=) მთლიანი სტრიქონების რაოდენობისა. ამ პარამეტრის არარსებობის შემთხვევაში, ფაილის ნაგულისხმევად მიიჩნევა პირველი სტრიქონი. first\_row შეიძლება იყოს დადებითი მთელი რიცხვი, რომლის მნიშვნელობა არ აღემატება  $2^{63}-1$ -ს და იწყება 1-დან.

**-G** - ეს გადამრთველი გამოიყენება Azure SQL Database ან Azure SQL Data Warehouse-თან მიერთებისას Azure Active Directory-ს ავთენტიფიკაციის გამოყენებით. **-G**-ს გამოყენებისათვის საჭიროა ვერსია [14.0.3008.27](#) ვერსია ან უფრო ახალი (ვერსიის დასადგენად შეასრულეთ **bcp -v** (AAD ინტეგრირებული და ინტერაქტიული ავთენტიფიკაცია ამჟამად არ არის მხარდაჭერილი Linux-სა და macOS-ში). იმისთვის, რომ შემოწმდეს, მხარდაჭერილია თუ არა არსებული bcp ვერსიაში Azure Active Directory (AAD) უნდა იქნეს შეყვანილი „**bcp -**“ (bcp <space> <dash> <dash>) და ხელმისაწვდომი არგუმენტების სიაში უნდა იყოს **-G**.

განვიხილოთ მაგალითები Azure Active Directory-ს ავთენტიფიკაციის გამოყენებით: Azure Active Directory-ს მომხმარებლის სახელისა და პაროლის გამოსაყენებლად შესაძლებელია მიეთითოს **-G** პარამეტრი და ასევე პარამეტრები **-U** (მომხმარებლის სახელი) და **-P** (პაროლი).

**მაგალითი 1.** მომხმარებლის სახელით და პაროლით: განხორციელდეს SQL Server-ის aadserver.database.windows.net მონაცემთა ბაზის testdb მონაცემთა ცხრილის bcptest ექსპორტი მონაცემთა ფაილში c:\last\data1.dat და გამოვიყენოთ Azure AD მომხმარებლის სახელი და პაროლი, რომლებიც არიან AAD-ს სააღრიცხვო მონაცემები:

```
bcp bcptest out "c:\last\data1.dat" -c -t -S aadserver.database.windows.net -d testdb  
-G -U alice@aadtest.onmicrosoft.com -P xxxxx
```

**მაგალითი 2.** მომხმარებლის სახელით და პაროლით: განხორციელდეს SQL Server-ზე aadserver.database.windows.net მონაცემთა ბაზაში testdb მონაცემთა ცხრილში bcptest იმპორტი მონაცემთა ფაილიდან c:\last\data1.dat და გამოვიყენოთ Azure AD მომხმარებლის სახელი და პაროლი, რომლებიც არიან AAD-ს სააღრიცხვო მონაცემები:

```
bcp bcptest in "c:\last\data1.dat" -c -t -S aadserver.database.windows.net -d testdb -G -U alice@aadtest.onmicrosoft.com -P xxxxx
```

**მაგალითი 3.** Azure Active Directory-ში ინტეგრირებული ავთენტიფიკაცია: Azure Active Directory ინტეგრირებული ავთენტიფიკაციისთვის საჭიროა გამოყენებულ იქნეს **-G** პარამეტრი მომხმარებლის სახელისა და პაროლის გარეშე (ამ კონფიგურაციაში იგულისხმება, რომ Windows-ის მომხმარებლის მიმდინარე ანგარიში, რომელშიც ხორციელდება bcp ბრძანება ჩართულია ფედერაციაში Azure AD-თან): მაგალითში განხორციელდეს ექსპორტი Azure AD-სთან ინტეგრირებული ანგარიშის გამოყენებით. ექსპორტირდება aadserver.database.windows.net SQL Server-ის testdb მონაცემთა ბაზის bcptest ცხრილი Azure AD ინტეგრირებული აუტენტიფიკაციის გამოყენებით მონაცემთა ფაილში c:\last\data2.dat:

```
bcp bcptest out "c:\last\data2.dat" -S aadserver.database.windows.net -d testdb -G -c -t
```



**მაგალითი 4.** განხორციელდეს იმპორტი Azure AD-სთან ინტეგრირებული ანგარიშის გამოყენებით. იმპორტირდება aadserver.database.windows.net SQL Server-ის testdb მონაცემთა ბაზის bcptest ცხრილში Azure AD ინტეგრირებული აუტენტიფიკაციის გამოყენებით მონაცემთა ფაილიდან c:\last\data2.dat:

```
bcp bcptest in "c:\last\data2.dat" -S aadserver.database.windows.net -d testdb -G -c -t
```

**მაგალითი 5.** Azure Active Directory-ში ინტერაქტიური ავთენტიფიკაცია: Azure Active Directory ინტერაქტიური ავთენტიფიკაციისთვის საჭიროა პაროლის ხელით შეყვანა გამოყენებულ იქნეს **-G** პარამეტრი მხოლოდ მომხმარებლის სახელით (-U), პაროლის გარეშე. მაგალითში ექსპორტი ხდება Azure AD ინტერაქტიული რეჟიმის გამოყენებით, მომხმარებლის სახელის მითითებით, სადაც მომხმარებელი წარმოადგენს AAD ანგარიშს. ინტერაქტიული რეჟიმისთვის საჭიროა პაროლის ხელით შეყვანა. მრავალფაქტორიანი ავთენტიფიკაციის მქონე ანგარიშებისთვის უნდა იქნეს გამოყენებული MFA-ს აუტენტიფიკაციის მეთოდი:

```
bcp bcptest out "c:\last\data1.dat" -c -t -S aadserver.database.windows.net -d testdb -G -U alice@aadtest.onmicrosoft.com
```



მაგალითი 6. Azure AD მომხმარებელი არის დომენის ფედერაციული მომხმარებელი სისტემა Windows ანგარიშით. ბრძანების სტრიქონში მომხმარებლის სახელი შეიცავს დომენის საადრიცხვო ჩანაწერს (მაგალითად, [joe@contoso.com](mailto:joe@contoso.com)):

```
bcp bcptest out "c:\last\data1.dat" -c -t -S aadserver.database.windows.net -d testdb  
-G -U joe@contoso.com
```

თუ სტუმარი მომხმარებლები არსებობენ კონკრეტულ Azure AD-ში და ისინი არიან ჯგუფის ნაწილი, რომელიც არსებობს SQL მონაცემთა ბაზაში და რომლებსაც აქვთ მონაცემთა ბაზის ნებართვები bcp ბრძანების შესასრულებლად, მაშინ გამოიყენება მათი სტუმრის მომხმარებლის ფსევდონიმი (მაგალითად, [\\*keith0@adventureworks.com](mailto:*keith0@adventureworks.com)).

გავაგრძელოთ პარამეტრების განხილვა.

**-h "load hints** [, ... n],,, - განსაზღვრავს ცხრილში მონაცემების დიდი მოცულობის იმპორტის დროს გამოყენებისათვის ერთს ან რამდენიმე მინიშნებას.

- **ORDER(column[ASC | DESC] [...n])** - მონაცემთა სორტირება მონაცემთა ფაილში. იმპორტის წარმადობა გაუმჯობესდება, თუ იმპორტირებული მონაცემები დალაგებულია ცხრილის კლასტერული ინდექსის მიხედვით (ასეთის არსებობის შემთხვევაში). თუ მონაცემთა ფაილის მონაცემები დალაგებულია კლასტერული ინდექსის გასაღებისაგან განსხვავებული თანმიმდევრობით, ან თუ მონაცემთა ცხრილს არ გააჩნია კლასტერული ინდექსი, ORDER პუნქტი იგნორირებულ უნდა იქნეს. დანიშნულების ცხრილში უნდა იქნეს მითითებული სვეტების სახელები. სტანდარტულად, bcp მიიჩნევს, რომ მონაცემთა ფაილში მონაცემები არ არიან დალაგებულნი. SQL Server ოპტიმიზირებული იმპორტისთვის ასევე ამოწმებს იმპორტირებული მონაცემების სორტირებას.
- **ROWS\_PER\_BATCH = bb** - მონაცემთა სტრიქონების რაოდენობა თითო პაკეტში (bb). გამოიყენება, როდესაც **-b** არ არის მითითებული, რის შედეგადაც მონაცემთა მთელი ფაილი SQL Server-ზე იგზავნება როგორც ერთი ტრანზაქცია. SQL Server-ი ახორციელებს მასიური დატვირთვის ოპტიმიზაციას bb მნიშვნელობის შესაბამისად. სტანდარტულად, ROWS\_PER\_BATCH უცნობია.

- **KILOBYTES\_PER\_BATCH = cc** - მონაცემების სავარაუდო კილობაიტების (CC) მოცულობა თითო პაკეტში. სტანდარტულად, KILOBYTES\_PER\_BATCH უცნობია.
- **TABLOCK** - განსაზღვრავს ბლოკირებას ცხრილის დონეზე მასიური ჩატვირთვის ოპერაციის განხორციელებისას (წინააღმდეგ შემთხვევაში ბლოკირება ხორციელდება მონაცემთა სტრიქონის დონეზე). ამ შემთხვევაში წარმადობა საგრძნობლად იზრდება, რადგან თითოეული სტრიქონისათვის არ ელოდება ბლოკირების რიგს. თუ ცხრილს არ აქვს ინდექსები და მითითებულია TABLOCK, მაშინ შეიძლება ერთდროულად ჩაიტვირთოს მრავალმა კლიენტმა. სტანდარტულად, დაბლოკვა განისაზღვრება ცხრილის პარამეტრით **table lock on bulkload**.
- **CHECK\_CONSTRAINTS** - სამიზნე ცხრილის ან წარმოდგენის ყველა შეზღუდვა უნდა შემოწმდეს მასიური იმპორტის ოპერაციის განხორციელებისას. CHECK\_CONSTRAINTS მინიშნების გარეშე, ნებისმიერი CHECK და FOREIGN KEY შეზღუდვები იგნორირებულია, ხოლო ოპერაციის შემდეგ ცხრილზე არსებული შეზღუდვა აღინიშნება, როგორც არასანდო (გამონაკლისია UNIQUE, PRIMARY KEY და NOT NULL, რომლებიც ყოველთვის მოწმდებიან). აქვე უნდა აღინიშნოს, რომ **-m max\_errors** პარამეტრი არ გამოიყენება შეზღუდვების შესამოწმებლად.
- **FIRE\_TRIGGERS** - გამოიყენება **in** არგუმენტთან ერთად და მიუთითებს, რომ მასიური კოპირებისას იმუშავებენ მიზნობრივი ცხრილის Insert ტრიგერები.



**-i input\_file** - ინტერაქტიულ რეჟიმის დროს მასიურ გადაწერისას განსაზღვრავს პასუხების ფაილის სახელს, რომელშიც მოცემულია მონაცემთა თითოეული სვეტისათვის ბრძანების სტრიქონის კითხვებზე პასუხები, თუ არა არის მითითებული -n, -C, -w, ან -N პარამეტრები. თუ *input\_file* იწყება დეფისით (-) ან წინა გადაკვეთით (/), არ უნდა იქნეს შეყვანილი სივრცე -i-სა და *input\_file* მნიშვნელობებს შორის.

**-k** - განსაზღვრავს, რომ ოპერაციის დროს ცარიელი სვეტები უნდა ინარჩუნებდნენ მნიშვნელობას Null და არა სვეტების რაიმე ნაგულისხმევ მნიშვნელობას.

**-K application\_intent** - აცხადებს პროგრამის დატვირთვის ტიპს SQL Server-თან მიერთებისას. ერთადერთი მნიშვნელობა, რაც შესაძლებელია, არის ReadOnly.

**-l login\_timeout** - განსაზღვრავს SQL Server- ში შესვლის შეყოვნებას (წამებში). შეყოვნების ნაგულისხმევი დროა 15 წამი (შესაძლებელია 0-დან 65534-მდე - სხვა შემთხვევაში bcp წარმოქმნის შეცდომის შეტყობინებას). 0 მნიშვნელობა განსაზღვრავს უსასრულო შეყოვნებას.

**-L last\_row** - განსაზღვრავს ბოლო სტრიქონის ნომერს (ცხრილიდან ექსპორტისათვის ან მონაცემთა ფაილიდან იმპორტისათვის). ამ პარამეტრის მნიშვნელობა უნდა იყოს მეტი (>) 0-ზე, მაგრამ ნაკლები (<) ან ტოლი (=) ბოლო სტრიქონის ნომერზე. ამ პარამეტრის არარსებობის შემთხვევაში, შეთანხმებით გამოიყენება ამ ფაილის ბოლო სტრიქონი. *last\_row* პარამეტრი შეიძლება იყოს დადებითი მთელი რიცხვი  $2^{63}-1$  მნიშვნელობამდე.



**-m max\_errors** - განსაზღვრავს სინტაქსური შეცდომების მაქსიმალურ რაოდენობას, რაც შეიძლება მოხდეს bcp ოპერაციის გაუქმებამდე (სინტაქსურ შეცდომაში იგულისხმება შეცდომა, რომელიც წარმოიშვება მონაცემთა გარდაქმნისას სამიზნე მონაცემების ტიპზე). max\_errors გამორიცხავს შეცდომებს, რომელთა დადგენა მხოლოდ SQL Server-ზე შეიძლება (მაგალითად, შეზღუდვების დარღვევა). მწკრივი, რომლის კოპირება შეუძლებელია bcp-ს გამოყენებით, იგნორირდება და ითვლება ერთ შეცდომად. თუ ეს პარამეტრი არ არის მითითებული, ნაგულისხმევია 10 მაქსიმალური შეცდომა.

**-n** - ახორციელებს მასობრივი კოპირების ოპერაციას საკუთარი მონაცემთა (მონაცემთა ბაზის) ტიპების გამოყენებით. ეს პარამეტრი არ ითხოვს თითოეულ სვეტისათვის მონაცემთა ტიპის განსაზღვრას, ის იყენებს საკუთარ მნიშვნელობებს.

**-N** - ახორციელებს მასობრივი კოპირების ოპერაციას საკუთარი მონაცემთა (მონაცემთა ბაზის) ტიპების გამოყენებით არასიმბოლური მონაცემთა ტიპებისათვის და უნიკოდის სიმბოლოებს სიმბოლური. ეს პარამეტრი უფრო მაღალი წარმადობის ალტერნატივაა, ვიდრე

**-w** და იგი არ ითხოვს მონაცემთა ტიპს თითოეულ სვეტისათვის. ამასთან ერთად იგი აუცილებლად უნდა იქნეს გამოყენებული ANSI გაფართოებულ სიმბოლოების (ნაციონალური ანბანის) არსებობისას მონაცემებში.

**-o output\_file** - განსაზღვრავს გამომავალი ფაილის სახელს. თუ output\_file იწყება დეფისით (-) ან წინა გადაკვეთით (/), არ შეიყვანოთ სივრცე **-o**-სა და output\_file მნიშვნელობებს შორის.

**-P password** - განსაზღვრავს შესვლის პაროლს ID-სათვის. თუ ეს პარამეტრი არ არის, bcp ბრძანება ითხოვს პაროლს. თუ ეს პარამეტრი გამოიყენება ბრძანების სტრიქონის ბოლოს პაროლის გარეშე, bcp იყენებს ნაგულისხმევ პაროლს (NULL). თუ მომხმარებელს სურს bcp ბრძანების შეყვანისას პაროლის დაფარვა, მაშინ ბრძანებაში არ უნდა მიეთითოს -P პარამეტრი -U-სთან ერთად და bcp-ს გაშვებისას (დააჭირეთ ENTER-ს) და ბრძანება მოითხოვს პაროლს. თუ პაროლი იწყება დეფისით (-) ან გადაკვეთის ხაზით (/), არ უნდა იქნეს შეყვანილი სივრცე -P და password მნიშვნელობას შორის.

**-q** - ახორციელებს SET QUOTED\_IDENTIFIERS ON ინსტრუქციას bcp-სა და SQL Server-ს შორის მიერთების დროს. ეს პარამეტრი გამოიყენება მონაცემთა ბაზის, მფლობელის, ცხრილის ან წარმოდგენის სახელის დასაზუსტებლად, რომელიც შეიცავს სივრცეს ან პწკარს (ერთმაგ ბრჭყალს). სახელი უნდა ჩაისვას ბრჭყალებში (""). -q არ ვრცელდება -d- ზე გადაცემულ მნიშვნელობებზე.

**-r row\_term** - განსაზღვრავს სტრიქონის დამთავრეს. ნაგულისხმევი არის \n (ახალი ხაზის სიმბოლო). თუ row\_term იწყება დეფისით (-) ან წინა ხაზი (/), არ უნდა იქნეს შეყვანილი სივრცე -r და row\_term მნიშვნელობებს შორის.

**-R** - განსაზღვრავს SQL Server-ში ვალუტის, თარიღისა და დროის მონაცემებს ისე, როგორც კლიენტის კომპიუტერის სისტემის რეგიონალურ ფორმატშია მითითებული რეგიონალურ ფორმატში. სტანდარტულად, რეგიონალური პარამეტრები იგნორირებულია.



**-S server\_name [\instance\_name]** - განსაზღვრავს SQL Server-ს, რომელთანაც ხორციელდება მიერთება bcp-თი. თუ SQL Server-ი არ არის მითითებული, bcp პროგრამა უერთდება ნაგულისხმევ SQL Server-ს (ადგილობრივ კომპიუტერში). ეს პარამეტრი საჭიროა, როდესაც bcp ბრძანება სრულდება დაშორებული კომპიუტერიდან ქსელში ან ლოკალური სახელობითი ინსტანციიდან.

**-t field\_term** - განსაზღვრავს ველის ბოლოს. ნაგულისხმევია \t (ტაბულაციის სიმბოლო). თუ field\_term იწყება დეფისით (-) ან წინა ხაზით (/), არ უნდა იქნეს შეყვანილი სივრცე -t-სა და field\_term მნიშვნელობებს შორის.

**-T** - განსაზღვრავს SQL Server-თან სანდო მიერთებას bcp-თი ინტეგრირებული უსაფრთხოების გამოყენებით. თუ -T არ არის მითითებული, მაშინ უნდა იქნეს მითითებული -U და -P. აქვე უნდა ითქვას, რომ როდესაც bcp უერთდება SQL მონაცემთა ბაზას ან SQL მონაცემთა საცავს, Windows ავთენტიფიკაციის ან Azure Active Directory ავთენტიფიკაციის გამოყენება არ არის მხარდაჭერილი. გამოიყენეთ -U და -P პარამეტრები.

**-U login\_id** - განსაზღვრავს შესვლის ID, რომელიც გამოიყენება SQL Server-თან მისაერთებლად.

**-v** - გამოსახავს bcp-ს ვერსიის ნომერს და საავტორო უფლებებს.

- V (80 | 90 | 100 | 110 | 120 | 130)** - ასრულებს მასიური კოპირების ოპერაციას SQL Server-ის ადრინდელი ვერსიის მონაცემთა ტიპების გამოყენებით. ეს პარამეტრი არ ითხოვს თითოეულ ტიპს, იგი იყენებს მნიშვნელობებს შეთანხმებით. აქ იგულისხმება (80 = SQL Server 2000 (8.x); 90 = SQL Server 2005 (9.x); 100 = SQL Server 2008 და SQL Server 2008 R2; 110 = SQL Server 2012 (11.x); 120 = SQL Server 2014 (12.x); 130 = SQL Server 2016 (13.x)).
- w** - ახორციელებს მასიური კოპირების ოპერაციას Unicode სიმბოლოების გამოყენებით. ეს პარამეტრი არ ითხოვს მონაცემთა ტიპს თითოეული ველისათვის, ის შენახვისას იყენებს nchar ტიპს, არ შეიცავს პრეფიქსებს, \t (ტაბულაციის სიმბოლო) გამოიყენება ველების გამყოფად, ხოლო \n-ს (ახალი სტრიქონის სიმბოლოს) როგორც როგორც მწკრივების დამთავრების. -w არ არის თავსებადი -C-თან.
- x** - გამოიყენება format და -f format\_file პარამეტრებით XML-ზე დაფუძნებულ ფორმატის ფაილის შესაქმნელად (ნაგულისხმევად იქმნება არა XML ფორმატის ფაილი). პარამეტრი -x არ მუშაობს მონაცემთა იმპორტის ან ექსპორტისას, რომლის დროსაც წარმოიშობა შეცდომა.



# 5 LEQC

საინფორმაციო სისტემა არის სტრუქტურირებული (მოწესრიგებული) მონაცემებისა და აპარატურულ-პროგრამული საშუალებების ერთობლიობა, განკუთვნილი მონაცემების შენახვისა და დამუშავებისათვის.

არსებობს საინფორმაციო სისტემების ორი კლასი:

- საინფორმაციო-საძებნი;
- მონაცემების დამუშავების.

საინფორმაციო-საძებნი საინფორმაციო სისტემა ორიენტირებულია საჭირო მონაცემების ძებნასა და მიღებაზე, ხოლო მონაცემების დამუშავების საინფორმაციო სისტემა - მონაცემების დამუშავებასა და დამუშავებული ინფორმაციის მიღებაზე. საინფორმაციო სისტემები ასრულებენ შემდეგ ფუნქციებს:

- ინფორმაციის შეტანა, ცვლილება და შენახვა;
- ინფორმაციის ნახვა და ძებნა;
- ინფორმაციის ამოღება სათანადო კრიტერიუმის მიხედვით;
- ანგარიშების ფორმირება;
- ინფორმაციის სისწორის შემოწმება.

## რისთვის არის საჭირო SQL-ი:

- მონაცემთა ახალი ბაზების, ცხრილებისა და წარმოდგენების შექმნისათვის;
- მონაცემთა ბაზების ცხრილებში ახალი ჩანაწერის ჩასმისათვის, კორექტირებისათვისა და წაშლისათვის;
- მონაცემთა ბაზებიდან მონაცემების ამოღებისთვის.

## ამისათვის SQL-ი აკეთებს:

- მოთხოვნას მონაცემთა ბაზებში მრავალი მეთოდის გამოყენებით;
- მონაცემთა ბაზების მართვის სისტემების გამოყენებით მონაცემთა ფორმატირებას;
- მონაცემთა აღწერას;
- მონაცემთა განსაზღვრასა და საჭიროების შემთხვევაში მანიპულირებას;
- მონაცემთა ბაზების, ცხრილებისა და მონაცემთა შექნას, წაშლასა და კორექტირებას;
- წარმოდგენის, პროცედურის, ფუნქციისა და ტრიგერის შექმნას;
- ცხრილებისათვის, პროცედურებისათვის და წარმოდგენებისათვის შეზღუდვების დაწესებას.

ფართო აზრით, მონაცემთა ბაზა არის სათანადო საგნობრივი სფეროს ობიექტების შესახებ მონაცემების მოწესრიგებული ერთობლიობა. საგნობრივი სფერო შეიძლება იყოს უნივერსიტეტი, საავადმყოფო, ფირმა და ა.შ., ობიექტი კი შესაბამისად, სტუდენტი, ავადმყოფი, თანამშრომელი და ა.შ.

პროგრამული საშუალებების ერთობლიობით მონაცემთა ბაზასთან მუშაობა შეიძლება მონაცემთა ბაზის მართვის სისტემით, რომელიც საჭიროა მონაცემთა ბაზის შესაქმნელად და მასში მოთავსებულ ობიექტებზე მანიპულირებისათვის.

არსებობს მონაცემთა განაწილებული (კლასტერული) და ცენტრალიზებული (არაკლასტერული) ბაზები. მონაცემთა განაწილებული ბაზა მოთავსებულია კომპიუტერული ქსელის რამდენიმე კომპიუტერზე (სერვერზე), ხოლო მონაცემთა ცენტრალიზებული ბაზა მოთავსებულია ერთ კომპიუტერზე, რომელზეც მიმართვა ექნებათ სხვა კომპიუტერებს კომპიუტერული ქსელის გამოყენებით.

მონაცემთა ბაზები არქიტექტურის მიხედვით ორგვარია: ფაილ-სერვერი და კლიენტ-სერვერი. ფაილ-სერვერული არქიტექტურის შემთხვევაში მონაცემთა ბაზა განთავსებულია ერთ ან მეტ ფაილ-სერვერზე, რომელიც წარმოადგენს ქსელში ჩართულ მძლავრ კომპიუტერს. მონაცემების დამუშავება სრულდება ლოკალურ კომპიუტერებზე. ფაილ-სერვერული მონაცემთა ბაზების მართვის სისტემებია: Microsoft Visual FoxPro, Microsoft Access, Paradox for Windows, dBase for Windows და ა.შ.



კლიენტ-სერვერული არქიტექტურის შემთხვევაში მონაცემთა ბაზა განთავსებულია სერვერზე და აქვე ხდება მისი დამუშავება. კლიენტის მხრიდან მონაცემების დამუშავების მოთხოვნა სერვერს ეგზავნება. მასზე სრულდება მონაცემების დამუშავება და შედეგები კლიენტს ეგზავნება. სისტემა, რომელიც იყენებს კლიენტ-სერვერულ ტექნოლოგიას ორი ნაწილისაგან შედგება: **კლიენტის ნაწილი (front-end)** და **სერვერის ნაწილი (back-end)**.

კლიენტის ნაწილი უზრუნველყოფს გრაფიკულ ინტერფეისს და იმყოფება მომხმარებლის კომპიუტერზე, ხოლო სერვერის ნაწილი მოთავსებულია სერვერზე და უზრუნველყოფს მონაცემების მართვას, დანაწილებას, ადმინისტრირებასა და უსაფრთხოებას. კლიენტ-სერვერული მონაცემთა ბაზების მართვის სისტემებია: Microsoft SQL Server, Oracle, IBM DB2, Sybase და ა.შ. ამ არქიტექტურისათვის დამახასიათებელია **მოთხოვნების სტრუქტურირებული ენის (SQL, Structured Query Language)** გამოყენება.

აქვე უნდა აღინიშნოს, რომ SQL არის მხოლოდ მოთხოვნების დამუშავების ენა, რომლის მუშაობისათვის საჭიროა ნებისმიერი მონაცემთა ბაზის დაყენება (Oracle, MySQL, MongoDB, PostgreSQL, SQL Server, DB2 და სხვ.). რაც შეეხება დასახელებას საერთაშორისო საზოგადოებაში მას უფრო See-Qwell-ს უძახიან.

SQL-ს როგორც მონაცემთა ბაზების სტანდარტულ ენას მოიხმარენ ისეთი მონაცემთა ბაზების მართვის სისტემები, როგორიცაა: Oracle, MySQL, MongoDB, PostgreSQL, SQL Server, DB2, MS Access, Sybase, Informix და სხვა.

მონაცემთა ბაზების კლასიფიცირება ხორციელდება მონაცემების მოდელების მიხედვით. მონაცემების მოდელი არის მონაცემების სტრუქტურისა და მათი დამუშავების ოპერაციების ერთობლიობა. მონაცემების მოდელის მიხედვით შესაძლებელია ობიექტების სტრუქტურისა და მათ შორის არსებული კავშირების წარმოდგენა. არსებობს მონაცემების შემდეგი მოდელები: იერარქიული, ქსელური და რელაციური. შესაბამისად, არსებობს იერარქიული, ქსელური და რელაციური მონაცემთა ბაზები.

მონაცემების იერარქიული მოდელის თითოეული ელემენტი სხვა ელემენტებს წარმოქმნის. წარმოქმნილი ელემენტები, თავის მხრივ, სხვა ელემენტებს წარმოქმნის და ა.შ. ამასთანავე, ნებისმიერ წარმოქმნილ ელემენტს აქვს მხოლოდ ერთი წარმომქმნელი ელემენტი. იერარქიული სტრუქტურის მაგალითია უნივერსიტეტი, საავადმყოფო და ა.შ. მონაცემების ასეთი ორგანიზების ნაკლია ის, რომ შეუძლებელია შემუშავებული იერარქიის ცვლილება და მოგვიწევს მხოლოდ იმ იერარქიის გამოყენება, რომელიც შექმნილი იყო მონაცემთა ბაზის დაპროექტების დროს. ამიტომ დადგა უფრო ზოგადი - ქსელური მოდელის შექმნა.

მონაცემების ქსელური მოდელის გამოყენებით წარმოქმნილ ელემენტს შეიძლება ჰქონდეს ერთზე მეტი წარმომქმნელი ელემენტი. ქსელური სტრუქტურის მაგალითია მაღაზია, სუპერმარკეტი და ა.შ. მაღაზიის გამყიდველი ემსახურება რამდენიმე მომხმარებელს. ამასთან, თითოეულ მომხმარებელს შეიძლება მოემსახუროს რამდენიმე გამყიდველი. ქსელურ მოდელს აქვს ნაკლი - იმისათვის, რომ მონაცემები დავამუშავოთ, უნდა ვიცოდეთ მონაცემთა ბაზის სტრუქტურა.



# მონაცემთა რელაციური ბაზა

მონაცემთა რელაციური მოდელის შემთხვევაში მონაცემების წარმოდგენა ხდება ორგანზომილებიანი ცხრილის სახით. მარტივ შემთხვევაში, რელაციური მოდელი აღწერს ერთ ორგანზომილებიან ცხრილს, უფრო ხშირად კი - რამდენიმე ცხრილის სტრუქტურასა და მათ შორის კავშირებს. რელაციური მოდელი ეფუძნება მათემატიკის ორ განყოფილებას: სიმრავლეთა თეორიასა და ლოგიკას (პრედიკატების აღრიცხვას).

მონაცემთა რელაციური ბაზების თეორია წინა საუკუნის სამოცდაათიან წლებში შეიმუშავა ამერიკელმა მათემატიკოსმა ე. კოდმა. მან მონაცემების დამუშავებისათვის შემოგვთავაზა სიმრავლეთა თეორიის აპარატი და დაამტკიცა, რომ მონაცემთა ნებისმიერი ნაკრები შეიძლება ორგანზომილებიანი ცხრილის სახით წარმოვადგინოთ. სიტყვა „რელაციური“ წარმოდგება ინგლისური სიტყვისაგან „relation“, რაც მიმართებას ნიშნავს. მიმართებების წარმოდგენა მოხერხებულია ორგანზომილებიანი ცხრილების სახით. ამრიგად, რელაციური არის მონაცემთა ისეთი ბაზა, რომელშიც მონაცემები წარმოდგენილია სწორკუთხა ორგანზომილებიანი ცხრილებით.

ცხრილს აქვს სახელი, რომელიც უნიკალურია მონაცემთა ბაზის შიგნით. ის შედგება სვეტებისა (ველებისა) და სტრიქონებისაგან (ჩანაწერებისაგან). ცხრილი შეიცავს ინფორმაციას ერთტიპური ობიექტების შესახებ, მისი სტრიქონი კი - ინფორმაციას კონკრეტული ობიექტის შესახებ. ცხრილის თითოეული სვეტი წარმოადგენს ობიექტების კონკრეტული ატრიბუტის მნიშვნელობების ერთობლიობას.

# მონაცემთა რელაციური ბაზა

სვეტს (ველს) აქვს სახელი და ტიპი, რომლის სახელიც უნიკალურია მონაცემთა ცხრილის შიგნით. თუმცა, განსხვავებულ ცხრილებში შეიძლება მოხმარებულ იქნეს ერთიდაიგივე სახელის მქონე სვეტები. ცხრილს უნდა გააჩნდეს ერთი სვეტი მაინც. ცხრილის სტრიქონებს სახელები არ აქვთ, მათი რიგითობა განსაზღვრული და რაოდენობა შეზღუდული არ არის.

მონაცემთა რელაციური ბაზა არის ერთმანეთთან ლოგიკურად დაკავშირებული ცხრილების ერთობლიობა, რომლის დაპროექტებისას უნდა დავიცვათ შემდეგი წესები:

- მონაცემთა ბაზაში თითოეულ ცხრილს უნდა ჰქონდეს უნიკალური სახელი და უნდა შედგებოდეს ერთტიპური სტრიქონებისაგან;
- თითოეული ცხრილი შედგება სვეტებისა და მნიშვნელობების ფიქსირებული რაოდენობისაგან;
- სტრიქონის ერთ სვეტში მოთავსებული უნდა იყოს მხოლოდ ერთი მნიშვნელობა;
- ცხრილში არ უნდა იყოს ზუსტად ერთნაირი ორი სტრიქონი - სტრიქონები უნდა განსხვავდებოდნენ ერთი სვეტის მნიშვნელობით მაინც;
- ცხრილის შიგნით თითოეულ სვეტის სახელი უნდა იყოს უნიკალური;
- სვეტისთვის უნდა განისაზღვროს მასში მოთავსებული მონაცემების ტიპი
- მონაცემების დამუშავებისას უნდა შეიძლებოდეს მიმართვა ნებისმიერ სტრიქონთან ან სვეტთან.



# მონაცემთა რელაციური ბაზა

მონაცემთა რელაციური ბაზა უნდა აკმაყოფილებდეს ნორმალიზაციის წესებს:

1. მონაცემთა ბაზაში საჭირო მონაცემების შენახვის უზრუნველყოფა;
2. მონაცემების სიჭარბის გამორიცხვა;
3. ცხრილების რაოდენობის მინიმუმამდე დაყვანა.

**ცხრილების ნორმალიზება** არის მონაცემების წარმოდგენის პროცესი მარტივი ორგანოზომილებიანი ცხრილებით, რომელიც საშუალებას გვაძლევს თავიდან ავიცილოთ მონაცემების დუბლირება და არაწინააღმდეგობა.

ნორმალიზების მიზანია მონაცემთა ბაზის ისეთი პროექტის ფარგლებში ინფორმაციის ნებისმიერი ნაწილი შენახული იქნება მხოლოდ ერთხელ და ერთ ადგილზე, რაც გამორიცხავს ინფორმაციის სიჭარბეს. ეს კეთდება ადგილის ეკონომიის და შენახული მონაცემების წინააღმდეგობრიობის გამორიცხვისათვის მიზნით.

ცხრილი ითვლება ნორმალიზებულად თუ იგი აკმაყოფილებს ნორმალიზების ფორმის მოთხოვნებს. არსებობს ცხრილის ნორმალიზების ექვსი ფორმა. ყველაზე ხშირად, ნორმალიზების პირველი სამი ფორმა გამოიყენება, რადგან მათი რეალიზება საკმაოდ ადვილია და ნორმალიზების საკმარის დონეს უზრუნველყოფენ.

# ცხრილებს შორის კავშირები

ცხრილის ნორმალიზების შემდეგ მიიღება ცხრილები, რომლებიც უნდა იყვნენ ერთმანეთთან დაკავშირებულნი, რისთვისაც გამოიყენება პირველადი და გარე გასაღებები.

ძირითადად, ორ ცხრილს შორის კავშირის დამყარების დროს ერთი არის მთავარი (მშობელი), მეორე კი - დამოკიდებული (შვილობილი). კავშირის დასამყარებლად მთავარი ცხრილის პირველადი გასაღები უკავშირდება დამოკიდებული ცხრილის გარე გასაღებს. როცა მთავარ ცხრილში ავირჩევთ რომელიმე სტრიქონს, მაშინ დამოკიდებული ცხრილიდან ამოირჩევა შესაბამისი სტრიქონები.

ნორმალიზების მეორე ფორმიდან გამომდინარე, ნებისმიერ მონაცემთა ცხრილს გააჩნია ერთი ან მეტი სვეტი, რომლებიც ცალსახად განსაზღვრავენ თითოეულ სტრიქონს. ასეთ სვეტ(ებ)ს **პირველადი გასაღები (PRIMARY KEY)** ეწოდება, რომელიც უნდა აკმაყოფილებდეს შემდეგ მოთხოვნებს:

- **უნიკალურობა** - ცხრილში არ უნდა იყოს ორი ისეთი სტრიქონი, რომლებსაც პირველადი გასაღების ერთნაირი მნიშვნელობები აქვთ;
- **მინიმალურობა** - პირველად გასაღებში შემავალი სვეტებიდან ნებისმიერის გამოკლება იწვევს უნიკალურობის დარღვევას.

# ცხრილებს შორის კავშირები

პირველად გასაღებისათვის გასათვალისწინებელია რამდენიმე რჩევა:

- არ შეიძლება უნიკალური სვეტ(ებ)ის მნიშვნელობის ცვლილება, რადგან იგი გამოიწვევს უნიკალურობის რღვევას (მაგ., თუ უნიკალური სვეტი არის „გვარი“, პიროვნების გვარის შეცვლის შემთხვევაში დაიკარგება ინფორმაცია როგორც მთავარ, ასევე დამოკიდებულ ცხრილებში);
- როდესაც პირველადი გასაღები შედგება რამდენიმე სვეტისაგან, უმჯობესია დამატებით შეიქმნას ერთი სვეტი და ის განისაზღვროს როგორც უნიკალური გასაღები.

**გარე გასაღები (FOREIGN KEY)** იქმნება დამოკიდებულ ცხრილში, რომელიც მთავარი ცხრილს მიმართავს. იგი, ასევე, შეიძლება შედგებოდეს ერთი ან მეტი სვეტისაგან. გარე გასაღების მნიშვნელობა უნდა არსებობდეს მთავარ ცხრილში, ან უნდა იყოს განუსაზღვრელი. ასევე, მთავარი ცხრილის პირველადი გასაღების რომელიმე მნიშვნელობა შეიძლება არ იყოს დაკავშირებული დამოკიდებულ ცხრილის არც ერთ სტრიქონის მნიშვნელობასთან.



# ცხრილებს შორის კავშირები

ცხრილებს შორის კავშირის შექმნისას შესაძლებელია დაყენებულ იქნეს პირველადი გასაღების შეცვლა/წაშლისას მოქმედების ოთხი ვარიანტი:

- **უმოქმედოდ (No Action)** - მთავარ ცხრილში პირველადი გასაღების მნიშვნელობების შეცვლა/წაშლისას დამოკიდებულ ცხრილში არაფერი არ შეიცვლება (კავშირი იკარგება);
- **კასკადური (Cascade)** - მთავარ ცხრილში პირველადი გასაღების მნიშვნელობების შეცვლა/წაშლისას იგივე განხორციელდება დამოკიდებულ ცხრილის იგივე მნიშვნელობების მქონე სტრიქონებში (კავშირი არ იკარგება);
- **განუსაზღვრელობა (Set Null)** - მთავარ ცხრილში პირველადი გასაღების მნიშვნელობის შეცვლა/წაშლისას დამოკიდებულ ცხრილში გარე გასაღები იღებს განუსაზღვრელ (NULL) მნიშვნელობას (კავშირი იკარგება).
- **შეთანხმებით (Set Default)** - დასაშვებია მთავარ ცხრილში პირველადი გასაღების მხოლოდ იმ მნიშვნელობის შეცვლა/წაშლა, რომლებიც არ არის დაკავშირებული დამოკიდებულ ცხრილის სტრიქონებთან (კავშირი არ იკარგება).

პრაქტიკიდან გამომდინარე შეიძლება ჩაითვალოს ყველაზე მომგებიან ვარიანტად ცვლილებისას კასკადური (კასკადურად განხორციელდება ცვლილება ყველა დამოკიდებულ ცხრილში), ხოლო წაშლისას - შეთანხმებით (წაიშლება მხოლოდ ის სტრიქონები, რომლებსაც არ გააჩნიათ შესაბამისი მნიშვნელობის სტრიქონები ყველა დამოკიდებულ ცხრილში).



# ცხრილებს შორის კავშირები

ნორმალიზების მესამე ფორმაზე დაყვანის შემდეგ, მთავარი ცხრილის მონაცემების სრული ინფორმაციის სანახავად საჭირო გახდა კავშირის განხორციელება დამოკიდებულ ცხრილთან, რისთვისაც გამოიყენება პირველადი და გარე გასაღებები. არსებობს სამი ტიპის კავშირი მონაცემთა ცხრილებს შორის:

- **„ერთი-ერთთან“** - ერთი ცხრილის ნებისმიერი სტრიქონი დაკავშირებულია მეორე ცხრილის ერთ ან არცერთ სტრიქონთან. ასეთი შემთხვევა წარმოიქმნება, როდესაც ერთი დიდი ცხრილი დაიყოფა ორ ან რამდენიმე ცხრილად და ყველა ცხრილში უნიკალური გასაღები იქნება მთავარი ცხრილის უნიკალური გასაღები;
- **„ერთი-ბევრთან“** - მთავარი ცხრილის ნებისმიერი სტრიქონი დაკავშირებულია დამოკიდებული ცხრილის არცერთ, ერთ ან მეტ სტრიქონთან, ხოლო დამოკიდებული ცხრილის ნებისმიერი სტრიქონი დაკავშირებულია მთავარი ცხრილის მხოლოდ ერთ სტრიქონთან. მონაცემთა ბაზებში უმეტეს შემთხვევაში ასეთი კავშირები წარმოიქმნება;
- **„ბევრი-ბევრთან“** - ერთი ცხრილის ნებისმიერი სტრიქონი დაკავშირებულია მეორე ცხრილის არცერთ, ერთ ან მეტ სტრიქონთან და პირიქითაც იგივეა. ასეთი კავშირების დროს შუაში უნდა იქნეს დამატებული ცხრილი, რომელთანაც ორივე ცხრილს ექნება კავშირი „ერთი-ბევრთან“.

# მონაცემთა ბაზების სტრუქტურები

SQL Server-ზე მონაცემები მონაცემთა ბაზებში ინახება. არსებობს მონაცემთა ბაზის ორი სტრუქტურა: ლოგიკური და ფიზიკური.

მონაცემთა ბაზის **ლოგიკური სტრუქტურა** მოიცავს: ცხრილების სტრუქტურას, მათ შორის კავშირებს, მომხმარებლების სიას, შენახულ პროცედურებს და მონაცემთა ბაზის სხვა ობიექტებს.

მონაცემთა ბაზის **ფიზიკური სტრუქტურა** მოიცავს: მონაცემთა ბაზის ფაილებისა და ტრანზაქციების ჟურნალის აღწერას, მათ საწყის ზომას, ნაზარდს ზომას, მაქსიმალურ ზომას, კონფიგურირების პარამეტრებს და სხვას.

# მონაცემთა ბაზების ობიექტები

მონაცემთა ბაზების ობიექტებია:

- **ცხრილები (Tables)** - ორგანზომილებიანი მატრიცებია, რომლებშიც უშუალოდ მონაცემები ინახება;
- **მონაცემთა ტიპები (Data Types)** - სისტემური მონაცემთა ტიპებია;
- **პირველადი (Primary Key) და გარე (Foreign Key) გასაღებები** - მონაცემთა ცხრილებს შორის კავშირების განსაზღვრისთვის სვეტები;
- **ცხრილებს შორის კავშირები** - მთავარი ცხრილის მონაცემების დამოკიდებულ ცხრილის მონაცემებთან კავშირში სრული ინფორმაციის სანახავად;
- **ნაგულისხმევი მნიშვნელობა (Default Value or Binding)** - მონაცემთა ბაზის ობიექტებია, რომლებიც შეგვიძლია მივუთითოთ უშუალოდ ცხრილის სტრუქტურაში ან მომხმარებლების მიერ განსაზღვრულ ტიპებში, რომლის მნიშვნელობაც აისახება მონაცემის არ შეტანის შემთხვევაში;
- **თვლადი სვეტი (Computed Column)** - მონაცემთა ცხრილის სტრიქონის სვეტში ამავე სტრიქონის სხვა სვეტებისაგან შემდგარი მათემატიკური მოქმედების ჩაწერა;



# მონაცემთა ბაზების ობიექტები

ასევე მონაცემთა ბაზების ობიექტებია:

- **წარმოდგენები (Views)** - ვირტუალური ცხრილებია, რომლებიც საშუალებას გვაძლევს ამოღების შედეგთან ვიმუშაოთ როგორც ცხრილთან;
- **ინდექსები (Indexes)** - სტრუქტურებია, რომლებიც ზრდის მონაცემებთან მუშაობის მწარმოებლურობას;
- **ტრიგერები (Triggers)** - სპეციალური შენახული პროცედურებია, რომლებიც ცხრილში მონაცემების ცვლილების დროს გამოიძახება;
- **მომხმარებლების ფუნქციები (user-defined functions)** - მომხმარებლების მიერ შექმნილი ფუნქციებია;
- **შენახული პროცედურები (stored procedures)** - Transact\_SQL-ის ბრძანებების ნაკრებია, შენახული გარკვეული სახელით. მათთან მიმართვა შესაძლებელია სახელის საშუალებით;
- **მთლიანობის შეზღუდვები (constraints)** - ობიექტებია, რომლებიც უზრუნველყოფენ მონაცემების ლოგიკურ მთლიანობას და არ არსებობს ცხრილებისაგან დამოუკიდებლად.

# მონაცემთა ბაზების ობიექტები

მონაცემთა ბაზის ობიექტებიდან მხოლოდ ცხრილი შეიცავს საკუთრივ მონაცემებს. როგორც ითქვა რელაციურ მონაცემთა ბაზებში არის ორ განზომილებიანი მონაცემთა ცხრილები, რომლებიც შედგებიან სტრიქონებისა და სვეტებისაგან:

- **თითოეული სტრიქონი (row)** წარმოადგენს კონკრეტული ობიექტის მახასიათებლების ერთობლიობას. მაგალითად, სტუდენტების ცხრილისათვის სტრიქონი წარმოადგენს კონკრეტული სტუდენტის მახასიათებლებს: გვარი, სახელი, უნივერსიტეტის დასახელება, კურსი და სხვ.
- **თითოეული სვეტი (column)** წარმოადგენს ობიექტების კონკრეტულ მახასიათებელს. მაგალითად, სტუდენტების ცხრილისათვის სვეტი წარმოადგენს სტუდენტების კონკრეტულ მახასიათებლებს: გვარი ან სახელი ან უნივერსიტეტის დასახელება ან კურსი ან სხვ. სვეტი არის ცხრილის მინიმალური ელემენტი და მას აქვს სახელი და ტიპი (შესაძლებელია ზომაც გარკვეული ტიპისათვის). ცხრილის ზოგიერთი სვეტი შეიძლება იყოს გამოთვლადი (computed). ასეთ სვეტებში ეთითება ფორმულა, რომლის მიხედვით სვეტის მნიშვნელობა გამოითვლება.

ცხრილის ყოველ სვეტს გააჩნია განსაზღვრული ტიპი, რომელიც მიუთითებს თუ როგორი ტიპის ინფორმაცია იქნება მოთავსებული ამ სვეტში.

# მონაცემთა ბაზების ობიექტები

ზოგჯერ საჭიროა მონაცემთა ცხრილში სტრიქონის დამატებისას გარკვეულ სვეტში თუ არ არის ცხადად მითითებული მნიშვნელობა ავტომატურად ჩაისვას მნიშვნელობა, რომელსაც **ნაგულისხმევი (ავტომატური) მნიშვნელობა (Default Value or Binding)** ეწოდება.

ნაგულისხმევი მნიშვნელობა შეიძლება იყოს მუდმივა, ჩადგმული ფუნქციის ან მათემატიკური გამოსახულების მიერ გადაცემული შედეგი. გასათვალისწინებელია, რომ ნაგულისხმევი მნიშვნელობა არ უნდა იყოს კონფლიქტში მთლიანობის შეზღუდვებთან.

ზოგ მონაცემთა ცხრილს ვერ უთითებთ პირველად გასაღებს, რადგან არ გააჩნია თავისი უნიკალური მონაცემი და ამ ცხრილთან მუშაობისას უნდა განხორციელდეს სვეტი-მთვლელის დამატება და განსაზღვრა, რომლისთვისაც შესრულდება უნიკალური მნიშვნელობების ავტომატურად გენერირება **IDENTITY** საკვანძო სიტყვის გამოყენებით. სვეტი-მთვლელის განსაზღვრისას უნდა მივუთითოთ მისი საწყისი მნიშვნელობა და ბიჯი (მითითების გარეშე სისტემა შეთანხმებით ორივეს მნიშვნელობას 1-ის ტოლს იღებს). სვეტისთვის IDENTITY თვისების განსაზღვრისას შემდეგი მოთხოვნები უნდა დავიცვათ:

- სვეტის ტიპი უნდა იყოს: BIGINT, DECIMAL, INT, NUMERIC, SMALLINT ან TINYINT;
- სვეტისთვის აკრძალული უნდა იყოს NULL მნიშვნელობის შენახვა;
- სვეტისთვის არ უნდა იყოს განსაზღვრული ავტომატური მნიშვნელობა (ნაგულისხმევი, თვლადი, ფუნქცია და სხვ.).



# მონაცემთა ბაზების ობიექტები

ზოგჯერ საჭიროა მონაცემთა ცხრილის სტრიქონის გარკვეულ სვეტში განხორციელდეს ამავე სტრიქონის სხვა სვეტებისაგან და მუდმივებისაგან შემდგარი მათემატიკური გამოსახულების ჩაწერა. ასეთ სვეტს **გამოთვლადი სვეტი (Computed Column)** ეწოდება. გასათვალისწინებელია, რომ გამოთვლადი სვეტის ფორმულაში შეიძლება გამოყენებულ იქნეს მხოლოდ მათემატიკური მოქმედებები.

**წარმოდგენები (Views)** არიან ვირტუალური ცხრილები, რომელიც შედგებიან სხვა ცხრილ(ებ)ის ან/და წარმოდგენ(ებ)ისაგან და ისინი განისაზღვრებიან SELECT მოთხოვნით. წარმოდგენა მონაცემებს არ შეიცავს - იგი გამოსახავს შემავალი ცხრილ(ებ)ის მონაცემებს. მონაცემები, რომლებიც გამოისახება ეკრანზე წარმოდგენის საშუალებით, ცალკე არ ინახება მონაცემთა ბაზაში. მარტივი წარმოდგენა იქმნება ერთი ცხრილის ბაზაზე და შეიძლება შეიცავდეს სვეტების ფილტრებს და სორტირებებს, ხოლო რთული წარმოდგენა შედგება რამდენიმე დაკავშირებული ცხრილის ან/და წარმოდგენის საფუძველზე. წარმოდგენასთან მიმართვა ხორცილდება ცხრილის ანალოგიურად სახელის გამოყენებით. წარმოდგენა გამოიყენება შემდეგ შემთხვევებში:

- მონაცემთა ცხრილ(ებ)ის გარკვეულ სტრიქონებთან/სვეტებთან მომხმარებლების მიმართვის შესაზღუდად;
- მონაცემთა დაკავშირებული ცხრილების მონაცემების ერთი ობიექტის სახით წარმოსადგენად;
- ინფორმაციის სანახავად, რომელიც მიიღება მონაცემების გარდაქმნის შედეგად.

# მონაცემთა ბაზების ობიექტები

**ინდექსები (Indexes)** არიან ცხრილთან ან წარმოდგენასთან დაკავშირებული სტრუქტურა და განეკუთვნებიან შესაბამის ცხრილში ან წარმოდგენაში ინფორმაციის ძებნის დასაჩქარებლად. ინდექსი განისაზღვრება ერთი ან მეტი სვეტისთვის, რომლებსაც ინდექსირებული სვეტები ეწოდებათ. ინდექსი შეიცავს ინდექსირებული სვეტის ან სვეტების დახარისხებულ მნიშვნელობებს საწყისი ცხრილის ან წარმოდგენის შესაბამის სტრიქონზე მიმართვებთან ერთად. მწარმოებლურობის ზრდა მიიღწევა სვეტის მონაცემების დახარისხების ხარჯზე.

**ფუნქცია** არის კონსტრუქცია, რომელიც შეიცავს ხშირად გამოყენებად კოდს და გააჩნია სახელი. ფუნქცია მონაცემებზე გარკვეულ მოქმედებებს ასრულებს და აბრუნებს კონკრეტულ წინასწარ განსაზღვრულ მნიშვნელობას. მისი გამოძახება მონაცემთა ცხრილისა და წარმოდგენის ანალოგიურად სახელით ხორციელდება ფრჩხილებში პარამეტრების მითითებით. არსებობს მომხმარებლის ფუნქციების ორი ტიპი:

1. ჩადგმული ფუნქციები (Built-in Functions) - სერვერის გარემოს შემადგენელი ნაწილია და სისტემის მიერ წინასწარ განსაზღვრულ მოქმედებებს ასრულებს (მაგ., MAX(), SUM() და სხვ.);
2. მომხმარებლის ფუნქციები (User-Defined Functions) - პროგრამირების გარემოს შემადგენელი ნაწილია და მომხმარებლის მიერ წინასწარ განსაზღვრულ მოქმედებებს ასრულებს.



# მონაცემთა ბაზების ობიექტები

**ტრიგერები (Triggers)** არიან შენახული პროცედურის სპეციალური კლასი, რომლებიც ავტომატურად გაიშვება ცხრილებში მონაცემების დამატების, ცვლილების ან წაშლის დროს. ტრიგერები იყოფიან სამ კატეგორიად: ჩამატების ტრიგერები (INSERT TRIGGER) - მუშაობას იწყებს მონაცემთა ცხრილში ახალი სტრიქონის დამატების შემდეგ; ცვლილების ტრიგერები (UPDATE TRIGGER) - მუშაობას იწყებს მონაცემთა ცხრილში არსებული სტრიქონის ჩასწორების შემდეგ; წაშლის ტრიგერები (DELETE TRIGGER) - მუშაობას იწყებს მონაცემთა ცხრილში არსებული სტრიქონის წაშლის შემდეგ. ერთ ტრიგერში შესაძლებელია იყოს როგორც ცალ-ცალკე ჩამატების, ცვლილებისა და წაშლის, ასევე მათი კომბინაციის მქონეც. ერთი ცხრილისთვის შესაძლებელია განსაზღვრული იყოს თითოეული ტიპის რამდენიმე ტრიგერი. გარდა ამისა, ერთი ტრიგერიდან შესაძლებელია სხვა ტრიგერების გამოძახება - ასეთ შემთხვევაში გვაქვს ჩადგმული ტრიგერები (nested triggers).

**შენახული პროცედურები (Stored Procedures)** არიან ერთ მოდულში გაერთიანებული ბრძანებების ჯგუფი, რომელსაც აქვს სახელი. ეს ბრძანებები კომპილირდება და სრულდება, როგორც ერთი მთლიანი. SQL Server-ზე ინახება სისტემური შენახული პროცედურების დიდი რაოდენობა. მათი სახელები იწყება „sp\_“ პრეფიქსით. ისინი გამოიყენება SQL Server-ის კონფიგურირებისა და მართვისათვის, სისტემურ მონაცემთა ბაზებსა და ცხრილებში მონაცემების შესაცვლელად და ა.შ. შესაძლებელია საკუთარი შენახული პროცედურის შექმნა, რომელიც მოთავსებული იქნება არსებულ მონაცემთა ბაზაში. შენახული პროცედურის შესასრულებლად გამოიყენება მისი სახელი და საჭიროების შემთხვევაში მიეთითება პარამეტრები.



# მონაცემთა ბაზების ობიექტები

**მთლიანობის შეზღუდვები (Constraints)** უზრუნველყოფენ ავტომატურ კონტროლს ჩვენ მიერ განსაზღვრულ პირობების/შეზღუდვების მონაცემების შესაბამისობაში ლოგიკურ დონეზე. მთლიანობის შეზღუდვები შეიძლება გამოყენებული იყოს სვეტის ან ცხრილის დონეზე. მთლიანობის შეზღუდვები, რომლებიც სვეტის დონეზე გამოიყენება, მოქმედებს მხოლოდ ამ სვეტში შესატან მონაცემებზე. თუ მთლიანობის შეზღუდვა გამოიყენება რამდენიმე სვეტის მიმართ, მაშინ შეზღუდვა მუშაობს ცხრილის დონეზე. ფუნქციურობის და გამოყენების მიხედვით არსებობს მთლიანობის შეზღუდვის ხუთი ტიპი:

1. NULL (უცნობი) მთლიანობის შეზღუდვა. ის მოქმედებს სვეტისა და მომხმარებლის ტიპის დონეზე. მათთვის შეგვიძლია განვსაზღვროთ NULL ან NOT NULL მთლიანობის შეზღუდვა;
2. CHECK (შემოწმება) მთლიანობის შეზღუდვა მოქმედებს სვეტის დონეზე და ზღუდავს სვეტში შესანახი მნიშვნელობების დიაპაზონს. მისი განსაზღვრისას შესატანი მონაცემებისათვის ეთითება ლოგიკური პირობა. მონაცემების შეტანის ან ჩამატების დროს შესატანი მნიშვნელობა მოთავსდება პირობაში. თუ შემოწმების შედეგია TRUE (ჭეშმარიტი), მაშინ მონაცემების ცვლილება ნებადართული იქნება, ხოლო თუ შემოწმების შედეგია FALSE (მცდარი), მაშინ ცვლილებები აიკრძალება და გაიცემა შეტყობინება შეცდომის შესახებ. ერთი სვეტისთვის შეგვიძლია შევქმნათ რამდენიმე CHECK შეზღუდვა. მისი მითითება შეგვიძლია ცხრილის შექმნის დროს;

# მონაცემთა ბაზების ობიექტები

3. UNIQUE (უნიკალურობა) მთლიანობის შეზღუდვა მოქმედებს სვეტის დონეზე და იძლევა სვეტში მნიშვნელობების უნიკალურობის გარანტიას. ცხრილში არ იქნება ორი სტრიქონი, რომლებსაც მოცემულ სვეტში ერთნაირი მნიშვნელობები ექნებათ. პირველადი გასაღებისაგან განსხვავებით UNIQUE მთლიანობის შეზღუდვა უშვებს NULL მნიშვნელობის არსებობას;
4. PRIMARY KEY (პირველადი გასაღები) მოქმედებს სვეტის ან ცხრილის დონეზე. პირველადი გასაღები შეიძლება შედგებოდეს ერთი ან მეტი სვეტისაგან და არის სტრიქონის უნიკალური იდენტიფიკატორი ცხრილის ფარგლებში. თუ პირველადი გასაღები ერთი სვეტისაგან შედგება, მაშინ ამ სვეტისთვის უნდა იყოს დაყენებული UNIQUE შეზღუდვა, ხოლო თუ პირველადი გასაღები რამდენიმე სვეტისაგან შედგება, მაშინ თითოეულ სვეტში მნიშვნელობები შეიძლება გამოვლინდეს ანუ არ იყოს უნიკალური, მაგრამ უნიკალური უნდა იყოს ამ სვეტების მნიშვნელობების კომბინაცია თითოეული სტრიქონისათვის. პირველად გასაღებში შემავალი არც ერთი სვეტისათვის არ უნდა იყოს დაყენებული NULL შეზღუდვა. ცხრილში შეგვიძლია შევქმნათ მხოლოდ ერთი პირველადი გასაღები.
5. FOREIGN KEY (გარე გასაღები) იქმნება ცხრილის დონეზე. დამოკიდებული ცხრილის გარე გასაღები უკავშირდება მთავარი ცხრილის პირველად გასაღებს. დამოკიდებულ ცხრილში არ შეიძლება სტრიქონის ჩასმა, თუ გარე გასაღებს არ აქვს შესაბამისი მნიშვნელობა მთავარ ცხრილში.



**დროებითი ცხრილები (temporary tables)** განკუთვნილია მონაცემების დროებით შესანახად და მოთავსებულია tempdb სისტემურ მონაცემთა ბაზაში. დროებითი ცხრილები ავტომატურად იშლება შეერთების დახურვის ან SQL Server-ის გაჩერების შემთხვევაში. არსებობს ორი სახის დროებითი ცხრილი:

- **ლოკალური დროებითი ცხრილი (local temporary tables)**, რომლის სახელი ერთი # სიმბოლოთი იწყება (მაგ., #LDC) და ამ სიმბოლოს მითითება საკმარისია ლოკალური დროებითი ცხრილის შესაქმნელად. ასეთი ცხრილი ხილულია მხოლოდ იმ შეერთების შიგნით, რომელშიც ის იქმნება, ხოლო შეერთების დახურვისას ის ავტომატურად იშლება. თუ ლოკალური დროებითი ცხრილი შეიქმნა შენახული პროცედურის მუშაობისას, მაშინ ამ პროცედურიდან გამოსვლისას ის ავტომატურად წაიშლება.
- **გლობალური დროებითი ცხრილი (global temporary table)**, რომლის სახელი ## სიმბოლოებით იწყება (მაგ., ##GDC) და ორჯერ ამ სიმბოლოს მითითება საკმარისია გლობალური დროებითი ცხრილის შესაქმნელად. ასეთი ცხრილი ხილულია ნებისმიერი შეერთებიდან და განკუთვნილია მონაცემების გასაცვლელად სხვადასხვა პროგრამას შორის. გლობალური დროებითი ცხრილი არსებობს იმ შეერთების დასრულებამდე, რომელშიც ის შეიქმნა, ხოლო მისი წაშლა და ცვლილება შესაძლებელია ყველა შეერთებიდან.



# სისტემური მონაცემთა ბაზები

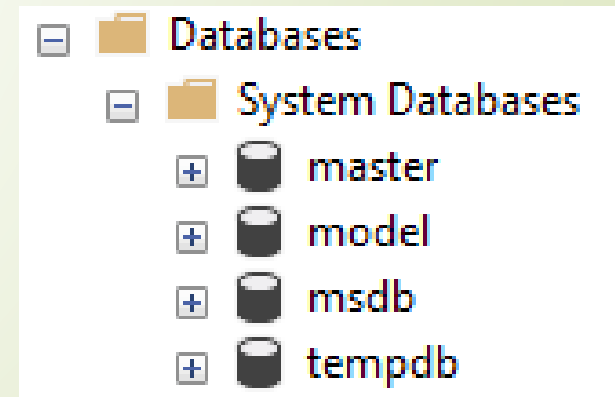
SQL Server-ს გააჩნია ოთხი ტიპის სისტემური მონაცემთა ბაზა:

- master;
- model;
- msdb;
- tempdb.

ეს მონაცემთა ბაზები გენერირდებიან SQL Server-ის ინსტალირებისას, ინახავენ სისტემურ მონაცემებს და ემსახურებიან შიგა ამოცანების გადაწყვეტას.

როგორც სისტემური, ისე მომხმარებლების მიერ შექმნილი მონაცემთა ბაზები, შეიცავენ სისტემურ ცხრილებსა და წარმოდგენებს. ასევე, ისინი შეიცავენ ინფორმაციას მონაცემთა ბაზის სტრუქტურაზე და მის ორგანიზებულობაზე.

სისტემურ ცხრილებთან მუშაობა UPDATE, INSERT და DELETE ბრძანებების საშუალებით აკრძალულია. დასაშვებია SELECT ბრძანების გამოყენება. ამ ცხრილებში მონაცემების შეცვლა შესაძლებელია მხოლოდ სისტემური შენახული პროცედურების გამოყენებით.



# სისტემური მონაცემთა ბაზები

master მონაცემთა ბაზა შეიცავს:

- სრულ სისტემურ ინფორმაციას SQL Server-ზე;
- ინფორმაციას SQL Server-ის სხვა მონაცემთა ბაზებზე;
- ინფორმაციას SQL Server-ის პირველადი ფაილების ადგილმდებარეობის შესახებ.

master მონაცემთა ბაზა შემდეგი ფაილებისაგან შედგება:

- master.mdf - მონაცემების ფაილი;
- mastlog.ldf - ტრანზაქციების ჟურნალის ფაილი.

ორივე ფაილი ინახება \Data კატალოგში (C:\Program Files\Microsoft SQL Server\...\MSSQL\Data\).

დაუშვებელია master სისტემურ მონაცემთა ბაზაში მომხმარებლის ობიექტების შექმნა.

# სისტემური მონაცემთა ბაზები

SQL Server-ზე ახალი მონაცემთა ბაზის შექმნა ხორციელდება მასში model სისტემური მონაცემთა ბაზის ობიექტების გადაწერის გზით.

model მონაცემთა ბაზა მოთავსებულია \Data კატალოგში და ორი ფაილისგან შედგება:

- ▶ model.mdf - მონაცემების ფაილი;
- ▶ model.ldf - ტრანზაქციების ჟურნალი.



# სისტემური მონაცემთა ბაზები

msdb სისტემური მონაცემთა ბაზა გამოიყენება SQL Server Agent სამსახურის მიერ მოვლენების (alerts), ამოცანებისა (jobs) და ოპერატორების (operators) რეგისტრირების დაგეგმვისათვის.

msdb მონაცემთა ბაზა ინახავს მთელ ინფორმაციას, რომელიც ეხება ადმინისტრირების ავტომატიზებასა და SQL Server-ის მართვას.

msdb ბაზა ორი ფაილისგან შედგება:

- msdbdata.mdf - მონაცემების ფაილი;
- msdblog.ldf - ტრანზაქციების ჟურნალი.

# სისტემური მონაცემთა ბაზები

სისტემური ცხრილები (**system tables**) შეიცავენ SQL Server-ის მუშაობისათვის საჭირო ინფორმაციას. ამიტომ, უფლება არ გვაქვს შეცვალოთ ამ ცხრილებში მოთავსებული მონაცემები.

აქედან გამომდინარე, ამ ცხრილების მიმართ შეგვიძლია მხოლოდ SELECT ბრძანების გამოყენება.

P.S. თუმცა აქვე უნდა აღვნიშნოთ, რომ შენახული პროცედურების გამოყენებით შეგვიძლია განვახორციელოთ მონაცემების ცვლილებები სისტემურ ცხრილებში.

# სისტემური მონაცემთა ბაზები

tempdb მონაცემთა ბაზა შეიცავს დროებით ობიექტებს, როგორიცაა ცხრილები, შენახული პროცედურები, ცვლადები, კურსორები და ა.შ. მასში, ინახება, როგორც სისტემური, ისე მომხმარებლის მიერ შექმნილი ობიექტები. SQL Server-ის ყოველი გაშვების დროს tempdb მონაცემთა ბაზა ხელახლა იქმნება და ყოველთვის იშლება SQL Server-ის გაჩერების დროს. მონაცემთა ბაზის ობიექტები ინახება მხოლოდ ერთი სეანსის განმავლობაში.

tempdb მონაცემთა ბაზა არის გლობალური კურსორი, რომელიც ავტომატურად მისაწვდომია ყველა მომხმარებლისათვის. შესაქმნელი დროებითი ობიექტები შეიძლება იყოს როგორც ლოკალური, ისე გლობალური. ლოკალური ობიექტი მისაწვდომია მხოლოდ მისი შემქმნელი მომხმარებლისთვის, გლობალური. კი - ყველა მომხმარებლისათვის. ლოკალური ობიექტები მოქმედებენ მხოლოდ სეანსის, შენახული პროცედურის, ტრიგერის ან ბრძანებების პაკეტის ფარგლებში. დროებითი ობიექტის შემქმნელი სტრუქტურიდან გამოსვლისას ეს ობიექტი მაშინვე წაიშლება, როგორც კი მომხმარებელი დაამთავრებს SQL Server-თან მუშაობას. ლოკალური დროებითი ცხრილის სახელი # სიმბოლოთი იწყება, გლობალური დროებითი ცხრილის სახელი კი - ## სიმბოლოებით.

tempdb მონაცემთა ბაზა ორი ფაილისაგან შედგება:

- tempdb.mdf - მონაცემების ფაილი;
- templogldf - ტრანზაქციების ჟურნალი.



# სისტემური მონაცემთა ბაზები

**დროებითი ცხრილები (temporary tables)** განკუთვნილია მონაცემების დროებით შესანახად და მოთავსებულია tempdb სისტემურ მონაცემთა ბაზაში. დროებითი ცხრილები ავტომატურად იშლება შეერთების დახურვის ან SQL Server-ის გაჩერების შემთხვევაში. არსებობს ორი სახის დროებითი ცხრილი:

- **ლოკალური დროებითი ცხრილი (local temporary tables)**, რომლის სახელი ერთი # სიმბოლოთი იწყება (მაგ., #LDC) და ამ სიმბოლოს მითითება საკმარისია ლოკალური დროებითი ცხრილის შესაქმნელად. ასეთი ცხრილი ხილულია მხოლოდ იმ შეერთების შიგნით, რომელშიც ის იქმნება, ხოლო შეერთების დახურვისას ის ავტომატურად იშლება. თუ ლოკალური დროებითი ცხრილი შეიქმნა შენახული პროცედურის მუშაობისას, მაშინ ამ პროცედურიდან გამოსვლისას ის ავტომატურად წაიშლება.
- **გლობალური დროებითი ცხრილი (global temporary table)**, რომლის სახელი ## სიმბოლოებით იწყება (მაგ., ##GDC) და ორჯერ ამ სიმბოლოს მითითება საკმარისია გლობალური დროებითი ცხრილის შესაქმნელად. ასეთი ცხრილი ხილულია ნებისმიერი შეერთებიდან და განკუთვნილია მონაცემების გასაცვლელად სხვადასხვა პროგრამას შორის. გლობალური დროებითი ცხრილი არსებობს იმ შეერთების დასრულებამდე, რომელშიც ის შეიქმნა, ხოლო მისი წაშლა და შეცვლა შეგვიძლია ყველა შეერთებიდან.

# ფაილები და ფაილების ჯგუფები

მონაცემთა ბაზის შესანახად სამი ტიპის ფაილი გამოიყენება:

- **Primary** - ძირითადი ფაილია, რომელიც შეიცავს სისტემურ ინფორმაციას თვით მონაცემთა ბაზისა და მისი ობიექტების შესახებ. მასში მოთავსებულია სისტემური ცხრილები და მონაცემთა ბაზის ობიექტების აღწერა. ძირითად ფაილში შეიძლება, აგრეთვე ინახებოდეს მონაცემებიც. ნებისმიერ მონაცემთა ბაზას აქვს primary ტიპის ფაილი, ამასთან მონაცემთა ბაზაში ამ ტიპის ფაილი მხოლოდ ერთი შეიძლება იყოს. primary ტიპის ფაილებს აქვთ .mdf (Master Data File ) გაფართოება.
- **Secondary** - მონაცემების არაძირითადი ფაილია, რომელიც არ შეიცავს სისტემურ ინფორმაციას და განკუთვნილია მხოლოდ მონაცემების შესანახად. მონაცემები, რომლებიც არ დაეტია ძირითად ფაილში, მოთავსდება მონაცემების არაძირითადი ფაილებში. მონაცემთა ბაზას შეიძლება საერთოდ არ ჰქონდეს secondary ტიპის ფაილი ან ჰქონდეს რამდენიმე. secondary ტიპის ფაილებს აქვთ .ndf (Not Master Data File) გაფართოება.
- **Transaction Log** - ტრანზაქციების ჟურნალის ფაილია. ის გამოიყენება მონაცემთა ბაზაში შესრულებული ტრანზაქციების შესახებ ინფორმაციის შესანახად. ნებისმიერ მონაცემთა ბაზას აქვს ტრანზაქციების ჟურნალის მინიმუმ ერთი ფაილი. transaction log ტიპის ფაილებს აქვთ .ldf (Log Data File) გაფართოება.

# ფაილები და ფაილების ჯგუფები

მონაცემთა ბაზაში გამოყენებულ თითოეულ ფაილს ორი სახელი აქვს:

1. ფაილის ლოგიკური სახელი (Logical File Name), რომელიც გამოიყენება Transact-SQL ბრძანებებში კონკრეტულ ფაილთან მიმართვისათვის.
2. ფაილის სახელი ოპერაციულ სისტემაში (OS File Name), რომელიც არის ფაილის ფიზიკური სახელი და ამ სახელით ინახება ფაილი დისკზე.

მონაცემთა ბაზის ყველა ფაილის ზომა შეიძლება ავტომატურად გაიზარდოს, რომელიც შესაძლებელია ფაილის შექმნის დროს განისაზღვროს. ნაზარდის ზომა შეგვიძლია განვსაზღვროთ პროცენტებში (ფაილის საწყისი ზომიდან) ან მეგაბაიტებში. დამატებით, შეგვიძლია მივუთითოთ ფაილის მაქსიმალური ზომა. თუ მაქსიმალური ზომა არ არის მითითებული, მაშინ ფაილის ზომა გაიზრდება მანამ, სანამ არის თავისუფალი სივრცე დისკზე.



# ფაილები და ფაილების ჯგუფები

ადმინისტრირებისა და მონაცემთა ბაზის ობიექტების ფიზიკური განლაგების მართვის გაადვილების მიზნით უმჯობესია ფაილები საფაილო ჯგუფებში მოვათავსოთ, რომლებიც არიან სამი ტიპის:

1. ფაილების ძირითადი ჯგუფი (Primary File Group) - შეიცავს primary ტიპის ფაილსა და ყველა ფაილს, რომლებიც არ არის ჩართული სხვა ჯგუფში (შეიძლება მხოლოდ ერთი ძირითადი ჯგუფი);
2. მომხმარებლების მიერ შექმნილი ფაილების ჯგუფი (User-defined File Group) - ჩართულია ყველა ფაილი, რომლებიც მითითებულია FILEGROUP პარამეტრში მონაცემთა ბაზის შექმნის (CREATE DATABASE) ან შეცვლის (ALTER DATABASE) დროს (შეიძლება რამდენიმე ჯგუფი ფაილების ნებისმიერი შემადგენლობით).
3. ფაილების ნაგულისხმევი ჯგუფი (Default File Group) - მონაცემთა ბაზაში ფაილების ერთ-ერთი ჯგუფი ხდება ნაგულისხმევი. თუ მონაცემთა ბაზის შექმნის დროს იგი არ არის მითითებული, მაშინ ფაილების ძირითადი ჯგუფი ხდება ნაგულისხმევი. თუ მონაცემთა ბაზის ობიექტის (ცხრილის ან სვეტის) შექმნის დროს აშკარად არ არის მითითებული, ფაილების რომელ ჯგუფს ეკუთვნის ის, მაშინ ეს ობიექტი შეიქმნება ფაილების ნაგულისხმევ ჯგუფში. მონაცემთა ბაზის მფლობელმა შეიძლება ფაილების ნებისმიერი ჯგუფი დანიშნოს ფაილების ნაგულისხმევ ჯგუფად. ფაილების მხოლოდ ერთი ჯგუფი შეიძლება იყოს ნაგულისხმევი.

## 6 LEQC

