

**SQL SERVERDA CHEKLASHLAR**

Tojimamatov Isroil Nurmamatovich

*Farg'ona davlat universiteti amaliy matematika va
informatika kafedrası katta o'qituvchisi*

israiltojimatov@gmail.com

Abdulhafizov Ibrohim Husanjon o'g'li

Farg'ona davlat universiteti 2-kurs talabasi

ibrohimabdulhafizov33@gmail.com

Annotatsiya: *Bu maqolada biz yangi boshlanuvchilar uchun juda qiziqarli, muhim va, ehtimol, qiyin mavzuga to'xtalamiz - Microsoft SQL Serverdagi cheklovlar. Ushbu maqolani o'qib bo'lgach, siz cheklovlar nima uchun kerakligini, SQL Serverda qanday turdagi cheklovlar mavjudligini va bu cheklovlarni qanday yaratishni bilib olasiz.*

Kalit so'zlar: *SQL Server, cheklashlar, PRIMARY KEY, FOREIGN KEY, NOT NULL, UNIQUE, CHECK, DEFAULT, ma'lumotlar yaxlitligi, bog'liqlik*

Аннотация: *В этой статье мы затронем очень интересную, важную и, возможно, сложную для новичков тему — ограничения Microsoft SQL Server. Прочитав эту статью, вы узнаете, зачем нужны ограничения, какие типы ограничений доступны в SQL Server и как их создавать.*

Ключевые слова: *SQL Server, ограничения, PRIMARY KEY, FOREIGN KEY, NOT NULL, UNIQUE, CHECK, DEFAULT, целостность данных, зависимость.*

Annotation: *In this article, we will touch on a very interesting, important and perhaps difficult topic for beginners - limitations in Microsoft SQL Server. After reading this article, you will know why constraints are needed, what types of constraints are available in SQL Server, and how to create these constraints.*

Keywords: *SQL Server, constraints, PRIMARY KEY, FOREIGN KEY, NOT NULL, UNIQUE, CHECK, DEFAULT, data integrity, dependency.*



Kirish

Cheklovlar - bu Microsoft SQL Serverdagi maxsus ob'ektlar bo'lib, ular ma'lumotlar bazasining avtomatik yaxlitligini ta'minlash uchun ustunlardagi ma'lum qiymatlarning haqiqiyliги to'g'risidagi qoidalarni belgilashga imkon beradi. Boshqacha qilib aytganda, cheklovlar jadvalga kiritiladigan va unda saqlanadigan ma'lumotlarga ma'lum bir shartni hosil qiladi. Masalan, ba'zi ma'lumotlar noyob bo'lishi kerak, turli jadvallardagi ba'zi ma'lumotlar umumiy kalit bilan uzviy bog'langan bo'lishi kerak va ba'zi ma'lumotlar umuman saqlanmasligi kerak, ya'ni har qanday ustunda hech qanday aniq qiymatlar bo'lmasligi kerak.

Cheklovlar, yuqorida aytib o'tganimdek, SQL Serverda juda muhim ob'ektlardir, chunki jadval tuzilishini rejalashtirish jarayonida siz doimo ma'lum bir ustunda saqlanishi mumkin bo'lgan maqbul qiymatlar haqida o'ylashingiz kerak. To'g'ri shaklda ma'lumotlarni qo'shish va saqlash jarayonini qo'lda boshqarish shunchaki mumkin emas. Shuning uchun SQL Server maxsus mexanizmni taqdim etadi, uning yordamida biz ma'lumotlarni kiritish va saqlash uchun qat'iy qoidalarni o'rnatishimiz mumkin, boshqacha qilib aytganda, ma'lumotlarning to'g'riligini kuzatish jarayonini avtomatlashtirishimiz mumkin. Agar siz ma'lumotlar bazasini loyihalash va amalga oshirishda cheklovlardan qochishni tanlasangiz, tez orada ma'lumotlar bazasi emas, balki tushunarsiz va nomuvofiq ma'lumotlar to'plamiga ega bo'lasiz.

SQL Serverdagi cheklovlar turlari

Microsoft SQL Server bir necha turdagi cheklovlarni amalga oshiradi, ularning har biri ma'lum bir vazifani bajarish uchun mo'ljallangan va endi biz bu turlarni ko'rib chiqamiz.

NOT NULL cheklovi

Bu biz taqiqlashimiz yoki aksincha, NULL qiymatlarni ustunda, ya'ni aniqlanmagan qiymatlarni saqlashga ruxsat berishimiz mumkin bo'lgan cheklovdir. Shunday qilib, biz aytishimiz mumkinki, agar biz ustunda NULL qiymatlarga ruxsat bermasak, unda bu ustunni to'ldirish majburiydir va agar biz NULL qiymatlarga ruxsat bersak, unda ustunni to'ldirish shart emas, ya'ni bu cheklov bizga ma'lum bir ob'ektning majburiy xususiyatlarini kiritish va saqlashni nazorat qilishimizga yordam beradi va



albatta, ma'lumotlar bazangizda NULL qiymatlarga ega bo'lish yaxshi narsa emasligini bilishingiz kerak, shuning uchun bu cheklov bunday qiymatlarni yo'q qilishga yordam beradi.

Ehtimol, siz ushbu cheklovga duch kelgansiz va u bilan ko'p marta ishlagansiz, chunki jadval yaratish yoki yangi ustun qo'shganda, biz deyarli har doim ustunning NULL qiymatlarini qabul qilish imkoniyatini ko'rsatamiz, buning uchun jadval ta'rifida NULL yoki NO NULL deb yozamiz.

PRIMARY KEY cheklovi

PRIMARY KEY – asosiy kalit cheklovi. Birlamchi kalit - qiymatlari har bir satrning o'ziga xosligini kafolatlaydigan ustun yoki ustunlar birikmasi bo'lib, bu bizga berilgan kalit orqali jadvaldagi har bir qatorni aniqlash imkonini beradi.

PRIMARY KEY deyarli har bir jadvalda bo'lishi kerak va unda faqat bitta bo'lishi kerak. Odatda, hisoblagich (IDENTITY) vazifasini bajaradigan va NULL qiymatlarni o'z ichiga olmaydi ustun uchun asosiy kalit yaratiladi. PRIMARY KEY cheklovini yaratish orqali siz birdaniga bir xil identifikatorga ega ikkita yozuvga ega bo'lgan jadvalingiz haqida tashvishlanishingiz shart emas.

FOREIGN KEY cheklovi

FOREIGN KEY — qo'shimcha kalit cheklovi. FOREIGN KEY cheklovi jadvallardagi ma'lumotlar o'rtasidagi munosabatni o'rnatish uchun mo'ljallangan. Boshqacha qilib aytganda, agar jadvalda boshqa jadvalda mavjud bo'lgan kalit (*ustun, odatda identifikator*) bo'lsa, bu jadvallar FOREIGN KEY cheklovi yordamida bog'lanishi kerak. Shunday qilib, ushbu cheklov yordamida biz ma'lumotlar bazasidagi jadvallar o'rtasida aloqa o'rnatamiz.

FOREIGN KEY cheklovi havolaning yaxlitligini ta'minlaydi, masalan, bir jadvalda boshqa jadvaldagi etishmayotgan yozuvlarga ishora qiluvchi yozuvlar mavjud bo'lgan, ya'ni bu yozuvlar yetishmayotgan va natijada noto'g'ri ma'lumotlarga olib keladigan vaziyatlarni istisno qilish imkonini beradi.

UNIQUE cheklovi

UNIQUE - bu ustun yoki ustunlar birikmasidagi qiymatlarning yagona bo'lishini ta'minlaydigan cheklovdir. UNIQUE ustundagi takroriy qiymatlarni yo'q



qilishga imkon beradi. PRIMARY KEY-dan farqli o'laroq, siz jadval uchun bir nechta UNIQUE cheklovlarni belgilashingiz mumkin va bu cheklov belgilangan ustun NULL qiymatini o'z ichiga olishi mumkin (*lekin siz tushunganingizdek, bu ustunda faqat bitta qiymat bo'lishi mumkin*). Agar jadval yaratilganda jadvaldagi ustunlar UNIQUE cheklovisiz aniqlangan bo'lsa, bu cheklovni qo'shish uchun tegishli ustunda takroriy qiymatlar bo'lmasligi kerak.

CHECK cheklovi

CHECK - chek cheklovi. Ushbu cheklash ma'lumotlar jadvalga kiritilganda ma'lum shartlarga javob berishini tekshiradi. Boshqacha qilib aytadigan bo'lsak, faqat ma'lum talablarga javob beradigan qiymatlarni saqlash uchun ustun kerak bo'lsa, ma'lumotlarni kiritishni boshqarish jarayonini avtomatlashtirish uchun CHECK cheklovidan foydalanishingiz mumkin. Masalan, biznes talabiga ko'ra, mahsulot narxi salbiy bo'lmasligi kerak, buning uchun mahsulot narxini saqlaydigan ustun uchun jadvalda biz ushbu ustunga kiritilgan barcha qiymatlarni tekshiradigan CHECK cheklovini belgilashimiz mumkin. Shunday qilib, biz ma'lum ustunlar uchun qanday qiymatlar qabul qilinishi haqida server darajasida aniq qoidalarni o'rnatamiz.

Jadvaldagi bitta ustunga bir nechta tekshirish cheklovlarini qo'llashimiz mumkin. Siz TRUE yoki FALSE deb baholanadigan har qanday mantiqiy ibora bilan UNIQUE cheklovini yaratishingiz mumkin.

DEFAULT cheklovi

DEFAULT - bu standart qiymat. Biz allaqachon NULL qiymati unchalik yaxshi emasligi haqida gapirgan edik, shuning uchun bu qiymatdan qutulishning yana bir usuli - ustun uchun standart qiymatni o'rnatish imkoniyati, agar biz ma'lumotlarni kiritishda hech qanday qiymat ko'rsatmasak, saqlanadi. Misol uchun, mahsulot narxi ko'rsatilgan ustunda narxni ko'rsatmasak, biz yangi mahsulot qo'shganimizda, SQL server avtomatik ravishda ushbu cheklovni belgilashda biz ko'rsatadigan standart qiymatni qo'shadi, masalan, 0.

NOT NULL va DEFAULT cheklovlarini yaratishga misol

NOT NULL va DEFAULT cheklovlarini yaratishga misol



Birinchidan, biz NOT NULL va DEFAULT cheklovlarini yaratish misolini ko'rib chiqamiz. Buni jadval yaratishda ham, undan keyin ham amalga oshirish mumkin, ya'ni cheklovni alohida ko'rsatma sifatida qo'shing.

```
CREATE TABLE TestTable(  
    [Column1] [INT] NOT NULL,  
    [Column2] [INT] NULL DEFAULT (0),  
    [Column3] [INT] NULL CONSTRAINT DF_C3 DEFAULT (0)  
) GO ALTER TABLE TestTable ALTER COLUMN [Column2] [INT] NOT  
NULL GO ALTER TABLE TestTable ADD CONSTRAINT DF_C1 DEFAULT (0)  
FOR Column1
```

Birinchi holat jadval yaratishda NOT NULL va DEFAULT cheklovlari qanday yaratilganligini ko'rsatadi.

Birinchi ustun uchun biz oddiygina NOT NULL ni belgiladik, ya'ni bu ustunda NULL qiymatlari bo'lishi mumkin emas.

Ikkinchi ustun uchun biz DEFAULT kalit so'zidan foydalangan holda standart qiymatni 0 ga o'rnatdik, bu cheklovni qo'shish uchun stenografiya.

Uchinchi ustun uchun biz standart qiymatni ham 0 ga o'rnatdik, lekin biz CONSTRAINT kalit so'zidan foydalangan holda to'liq cheklov ta'rifidan foydalandik.

Ikkinchi bayonot mavjud jadvalga NOT NULL cheklovini qanday qo'shishni ko'rsatadi. Ko'rib turganingizdek, bu ALTER TABLE operatori va ALTER COLUMN buyrug'i yordamida amalga oshiriladi. Dastlab, 2-ustun NULL qiymatiga ega bo'lishi mumkin edi, ammo bu buyruqni bajargandan so'ng u mumkin emas. Ammo shuni yodda tutish kerakki, agar ustun allaqachon NULL qiymatlarni o'z ichiga olgan bo'lsa, bayonot bajarilmaydi.

Uchinchi bayonotda Column1 ustuniga DEFAULT cheklovini qo'shdik, buning uchun biz ALTER COLUMN operatoridan ham foydalandik va cheklovni qo'shish uchun ADD CONSTRAINT buyrug'idan foydalandik, shundan so'ng biz cheklov nomini (DF_C1), turini yozdik va FOR kalit so'zidan foydalanib biz cheklov yaratmoqchi bo'lgan ustunni belgilab oldik.

Microsoft SQL Serverda PRIMARY KEY cheklovini yaratish misoli



PRIMARY KEY cheklovlari jadval yaratish paytida (ikki xil usulda) yoki alohida bayonotdan foydalangandan keyin yaratilishi mumkin.

Birinchidan, jadval yaratish jarayonida birlamchi kalit qanday yaratilishini ko'rib chiqamiz.

```
CREATE TABLE TestTable2(  
    [Column1] [INT] IDENTITY(1,1) NOT NULL CONSTRAINT  
    PK_Column1_T2 PRIMARY KEY,  
    [Column2] [VARCHAR](100) NOT NULL  
) GO CREATE TABLE TestTable3(  
    [Column1] [INT] IDENTITY(1,1) NOT NULL,  
    [Column2] [VARCHAR](100) NOT NULL,  
    CONSTRAINT PK_Column1_T3 PRIMARY KEY (Column1 )
```

Birinchi usul ustunlar darajasida PRIMARY KEY ni aniqlashni o'z ichiga oladi, ya'ni barcha xususiyatlardan so'ng biz CONSTRAINT kalit so'zini, keyin cheklov nomini va ushbu cheklov turini yozdik.

Ikkinchi yo'l - jadval darajasida cheklovni aniqlash, boshqacha qilib aytganda, barcha ustunlardan so'ng biz CONSTRAINT kalit so'zini, cheklash nomini, turini yozamiz va bu holda biz qaysi ustun asosiy kalit bo'lishini ham ko'rsatamiz (bizning holatda, 1-ustun).

Mavjud jadvalga asosiy kalit qo'shish uchun ALTER TABLE operatori va ADD CONSTRAINT buyrug'idan foydalanish kerak.

```
ALTER TABLE TestTable ADD CONSTRAINT PK_TestTable PRIMARY  
KEY (Column1)
```

Bu holda biz TestTable jadvaliga PK_TestTable deb nomlangan asosiy kalit cheklovini qo'shdik, 1-ustun asosiy kalit bo'ladigan ustundir.

SQL Serverda FOREIGN KEY cheklovini yaratish misoli

Shuningdek, biz FOREIGN KEY cheklovlarini jadvalni yaratishda ham, mavjud jadvalga tegishli alohida bayonot sifatida ham belgilashimiz mumkin.



Misol uchun, ikkita jadval tuzamiz, birinchisida mahsulot toifalari bo'yicha ma'lumotlar bo'ladi, ikkinchisida toifaga havolasi bo'lgan mahsulotlar ro'yxati bo'ladi, ya'ni mahsulot qaysidir toifaga tegishli bo'lishi kerak.

```
CREATE TABLE TestTable4(  
    [CategoryId] [INT] IDENTITY(1,1) NOT NULL,  
    [CategoryName] [VARCHAR](100) NOT NULL,  
    CONSTRAINT PK_TestTable4 PRIMARY KEY (CategoryId)  
)  
CREATE TABLE TestTable5(  
    [ProductId] [INT] IDENTITY(1,1) NOT NULL,  
    [CategoryId] [INT] NOT NULL,  
    [ProductName] [VARCHAR](100) NOT NULL,  
    [Price] [MONEY] NULL,  
    CONSTRAINT PK_TestTable5 PRIMARY KEY (ProductId),  
    CONSTRAINT FK_TestTable5 FOREIGN KEY (CategoryId)  
REFERENCES TestTable4 (CategoryId)  
ON DELETE CASCADE  
ON UPDATE CASCADE
```

)Bu misolda biz avval havola qiladigan jadvalni, ya'ni toifalar bilan jadvalni yaratdik, keyin mahsulotlar bilan jadval tuzdik va uni yaratishda chet el kalitini aniqladik, ya'ni FOREIGN KEY cheklovini yaratdik. Buning uchun, xuddi birlamchi kalitni yaratishda bo'lgani kabi, biz CONSTRAINT kalit so'zini, cheklash nomini, turini va boshqa jadvaldagi kalitga murojaat qiladigan ustunni ko'rsatdik. Keyinchalik, biz REFERANSLAR kalit so'zini yozdik, kalitni o'z ichiga olgan jadvalni ko'rsatdik va qavs ichida kalit bo'ladigan ustun nomini ko'rsatdik; ko'p hollarda, bu ustun asosiy kalit, lekin shart emas.

Misolda men kalitda o'chirish yoki yangilash operatsiyasi bajarilgan hollarda ba'zi harakatlarni belgilashimiz mumkinligini ham ko'rsatdim. Misol uchun, siz mahsulot toifasini o'chirishga qaror qildingiz, lekin bu yozuvga mahsulot jadvalidagi yozuvlar havola qilingan bo'lsa (*mahsulotlar ushbu turkumga bog'langan bo'lsa*) buni amalga oshira olmaysiz, boshqacha qilib aytganda, sukut bo'yicha xato bo'ladi.



Standart amalni o'zgartirish uchun cheklovni belgilaydigan bayonotda ON DELETE va ON UPDATE buyruqlarini, ya'ni mos ravishda kalit o'chirilganda bajariladigan amallarni va ushbu kalit yangilanganda bajariladigan amallarni belgilashimiz mumkin. Quyidagi qiymatlarni belgilash mumkin:

- CASCADE - o'chirilgan taqdirda, barcha havola qilingan yozuvlar o'chiriladi, ya'ni agar biz toifani o'chirmoqchi bo'lsak, biz ushbu turkumdagi barcha mahsulotlarni ham o'chirib tashlaymiz (SQL serveri buni avtomatik ravishda bajaradi). Yangilangan taqdirda barcha yozuvlar yangilanadi;
- SET NULL – o'chirishda ham, yangilashda ham NULL qiymatini belgilash;
- SET DEFAULT — o'chirishda ham, yangilashda ham standart qiymatni belgilash;
- HAKKAT YO'Q – hech narsa qilmang, shunchaki xatolik yuz beradi (ixtiyoriy, bu standart amal).

FOREIGN KEY cheklovi bitta bayonotda qanday yaratilganligini ko'rish uchun keling, mahsulotlar jadvalini o'chiramiz, so'ngra uni tashqi kalitsiz yaratamiz va keyin FOREIGN KEY cheklovini qo'shamiz.

DROP TABLE TestTable5

CREATE TABLE TestTable5(

[ProductId] [INT] **IDENTITY**(1,1) NOT NULL,

[CategoryId] [INT] NOT NULL,

[ProductName] [VARCHAR](100) NOT NULL,

[Price] [MONEY] NULL,

CONSTRAINT PK_TestTable5 **PRIMARY KEY** (ProductId)

)**ALTER TABLE** TestTable5 **ADD CONSTRAINT** FK_TestTable5
FOREIGN KEY (CategoryId)

REFERENCES TestTable4 (CategoryId);

FOREIGN KEY cheklovi mavjud jadvalga boshqa cheklovlar kabi ALTER TABLE operatori va ADD CONSTRAINT buyrug'i yordamida qo'shiladi. Ushbu



misolda, siz ko'rib turganingizdek, men ON DELETE va ON UPDATE iboralarini ko'rsatmadim, ya'ni o'chirish yoki yangilashda amal sukut bo'ladi.

MS SQL Serverda CHECK cheklovini yaratish misoli

CHECK cheklovlari ikkita usulda ham yaratilishi mumkin: jadval yaratish vaqtida va alohida ALTER TABLE bayonoti orqali.

```
CREATE TABLE TestTable6(  
    [Column1] [INT] NOT NULL,  
    [Column2] [INT] NOT NULL,  
    CONSTRAINT CK_TestTable6_C1 CHECK (Column1 <> 0)  
) ALTER TABLE TestTable6 ADD CONSTRAINT CK_TestTable6_C2  
CHECK (Column2 > Column1);
```

Birinchi holda, biz jadval yaratdik va unda CK_TestTable6_C1 chek cheklovini darhol aniqladik, bu esa Ustun1 ustunida 0 qiymatini o'z ichiga olmaydi. Boshqacha qilib aytganda, agar siz to'satdan siz yoki boshqa birov Column1 = 0 qiymatiga ega bo'lgan qatorni kiritmoqchi bo'lsa, SQL Server buni amalga oshirishga ruxsat bermaydi, u xatoni qaytaradi.

Ikkinchi holda, biz tekshirish cheklovini qo'shdik va ikkita ustundan foydalandik, ya'ni satrlarda 2-ustun qiymati har doim 1-ustundagi qiymatdan katta bo'lishi kerakligini aytamiz.

UNIQUE cheklash yaratish misoli

Biz bir necha usul bilan UNIQUE cheklovni yaratishimiz mumkin. Jadvalni ustunlar darajasida yaratishda, jadval darajasida jadval yaratishda va alohida ALTER TABLE ADD CONSTRAINT bayonoti bilan.

Keling, bu qanday amalga oshirilganini ko'rib chiqaylik.

```
CREATE TABLE TestTable7(  
    [Column1] [INT] NOT NULL CONSTRAINT U_TestTable7_C1  
UNIQUE,  
    [Column2] [INT] NOT NULL,  
    [Column3] [INT] NOT NULL,  
    CONSTRAINT U_TestTable7_C2 UNIQUE (Column2)
```



)ALTER TABLE TestTable7 ADD CONSTRAINT U_TestTable7_C3
UNIQUE (Column3);

Xulosa

Biz uchta ustundan iborat jadvalni yaratdik va biz ushbu ustunlarning barchasiga UNIQUE cheklovini turli yo'llar bilan qo'lladik.

Birinchi ustun uchun ustunning o'zi darajasida, ikkinchisi uchun alohida ko'rsatma, uchinchisi uchun stol darajasida.

FOYDALANILGAN ADABIYOTLAR:

1. Tojimamatov, I., & Mirsiddiqova, M. (2025). BERILGANLAR BAZASIDA HAYOTIY SIKL. *Модели и методы в современной науке*, 4(6), 66-70.
2. Tojimamatov, I., & Abduvaliyev, X. (2025). KO 'P FOYDALANUVCHILI BBBT ARXITEKTURASI. *Инновационные исследования в науке*, 4(5), 16-22.
3. Tojimamatov, I., & Siddiqova, G. (2025). TRANZAKSIYALARNI TAQSIMLANGAN TARZDA QAYTA ISHLASH MODELLARI. *Современные подходы и новые исследования в современной науке*, 4(6), 30-35.
4. Нурмамамович, Т. И., & Рахила, А. (2025). НА ОСНОВЕ МАТЕМАТИЧЕСКИХ МОДЕЛЕЙ ПОВЫШЕНИЕ УСТОЙЧИВОСТИ К ПОЛОМКАМ И АВАРИЯМ. *YANGI O 'ZBEKISTON, YANGI TADQIQOTLAR JURNALI*, 2(8), 197-204.
5. Тоджиматов И., Холмуродов А. О. Х. (2025, май). SQL-СЕРВЕРДА ЧЕКЛАШЛАР. В *КОНФЕРЕНЦИИ ПО СОВРЕМЕННОЙ НАУКЕ И ПЕДАГОГИКЕ* (Том 1, № 1, стр. 409-413).
6. Tojimamatov, I., & Ahmataliyeva, S. (2025). BERILGANLARNI MARKAZLASHGAN TARZDA BOSHQARISH TAMOYILLARI. *Академические исследования в современной науке*, 4(21), 59-64.
7. Tojimamatov, I., & Marifjonova, M. (2025). BERILGANLARNI MARKAZLASHGAN TARZDA BOSHQARISH PRINSIPLARI. *Академические исследования в современной науке*, 4(21), 65-70.



8. Tojimamatov, I., & Adxamova, C. (2025). AMALIY TIZIMLARDA BERILGANLAR BAZASINI BOSHQARISH TIZIMLARI O‘RNI. *Академические исследования в современной науке*, 4(21), 77-82.
9. Tojimamatov, I., & Fazliddinov, X. (2025). BERILGANLAR BAZASI ADMINISTRATORI VA UNING XUSUSYATLAR. *Академические исследования в современной науке*, 4(21), 90-95.
10. Tojimamatov, I. (2025). BERILGANLAR BAZASIDA AXBOROT ALMASHINISH SXEMASI HAMDA TURLARI. *Академические исследования в современной науке*, 4(21), 71-76.
11. Tojimamatov, I., & Zaylobiddinova, M. (2025). AUDIO MA'LUMOTLAR TAQDIMOT. *TRANSFORMING EDUCATION THROUGH SCIENTIFIC DISCOVERY*, 1(1), 38-44.
12. Tojimamatov, I., & Ismoilov, J. (2025). KRIPTOGRAFIK KALITLARNING KELAJAGI: XAVFSIZLIKNI TA'MINLASHDA YANGI YO‘NALISHLAR. *BRIDGING THE GAP: EDUCATION AND SCIENCE FOR A SUSTAINABLE FUTURE*, 1(1), 56-63.
13. Tojimamamtov, I. (2025). РАҚАМЛИ ИҚТИСОДИЁТНИНГ ИЖТИМОЙ-ИҚТИСОДИЙ ЖАРАЁНЛАРНИ БОШҚАРИШ ВА РЕЖАЛАШДАГИ МОҲИЯТИ. *DIGITAL TRANSFORMATION AND ARTIFICIAL INTELLIGENCE*, 3(1), 229-233.
14. Tojmamatov, I., Sharofutdinov, I., & Saydolimova, G. (2025). BILIMLAR OMBORI. BILIMLAR OMBORINING TUZILISHI. *EDUCATION AND RESEARCH IN THE ERA OF DIGITAL TRANSFORMATION*, 1(1), 431-437.
15. Tojimamatov, I., & Metinboyeva, F. (2025). TESKARI MUNOSABATDA MUNOSABATLAR USTIDAGI AMALLAR. *BRIDGING THE GAP: EDUCATION AND SCIENCE FOR A SUSTAINABLE FUTURE*, 1(1), 202-214.
16. Microsoft SQL Server Documentation: <https://learn.microsoft.com/en-us/sql/sql-server>
17. Itzik Ben-Gan, *T-SQL Fundamentals*, Microsoft Press, 2016.



18. Peter Rob & Carlos Coronel, *Database Systems: Design, Implementation, and Management*.
19. W3Schools – SQL Constraints:
https://www.w3schools.com/sql/sql_constraints.asp
20. StackOverflow – SQL Best Practices for Constraints:
<https://stackoverflow.com>