

1) A primary key is a unique identifier for each row (record) in a database table.

Oddiy qilib aytganda:

Primary key — bu har bir qatorni 100% noyob aniqlab beradigan ustun.

Asosiy xususiyatlari:

Noyob bo'ladi — ikki qator bir xil primary key qiymatga ega bo'la olmaydi

NULL bo'la olmaydi

Odatda: id, user_id, product_id kabi ustunlar bo'ladi

Har doim bitta qatordagi ma'lumotni topish, o'chirish, yangilash uchun ishlatiladi

Misol:

id (PK)	name	price
1	Olma	5000
2	Nok	7000
3	Banan	9000

Bu jadvalda id — primary key.

Bir jumlada ta'rif:

Primary key — jadvaldagi har bir yozuvni noyob aniqlab beradigan ustun

2) Power BI'da jadval munosabatlarining ikki asosiy turi mavjud:

☒ 1. One-to-Many (1: yoki 1-to-Many)*

Eng ko'p ishlatiladigan relationship.

Bitta jadvaldagi bitta qiymat ikkinchi jadvaldagi ko'p qiymatlar bilan bog'lanadi.

Masalan:

Category (1) → Products (many)

☒ 2. Many-to-Many (:)

Ikkala jadvalda ham takroriy qiymatlar bo'lishi mumkin.

Ko'pincha bridging table orqali ishlaydi.

3) Method 1: Using the Model View (most common)

Go to "Model" view

(left sidebar → the diagram icon)

You will see all your tables visually.

Click and drag the column from one table

→ to the matching column in the other table

(for example: CustomerID → CustomerID)

Power BI automatically opens the Create Relationship window.

Check:

Tables

Columns

Cardinality (1:*, *:1, :)

Cross filter direction (Single/Both)

Click OK.

☒ Method 2: Using Manage Relationships

Go to: Home → Manage Relationships

Click New

Select:

First table

First column

Second table

Second column

Choose relationship type:

One-to-Many

Many-to-Many

Click OK.

4) Star schema — bu markazda bitta fact jadvali, uning atrofida esa bir nechta dimension jadvallari bo'lgan sxema. Ko'rinishi yulduzga o'xshagani uchun shunday nomlangan.

1. Fact jadval (markazda)

Bu jadval raqamli qiymatlar, hisob-kitoblar va transaktsiyalarni o'z ichiga oladi:

sales_amount

quantity

total_price

revenue

clicks

orders

2. Dimension jadvallari (atrofida)

Bu jadvallar tavsifiy ma'lumotlarni saqlaydi:

Product (product_name, brand, category)

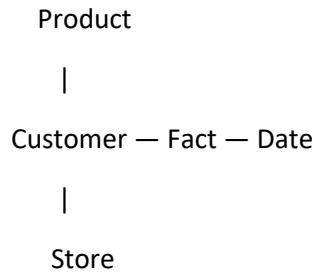
Customer (name, age, region)

Date (year, month, day)

Store (location, manager)

★ Nega “star” deyiladi?

Chizmasi yulduzga o‘xshaydi:



🎯 Star schema nima uchun ishlatiladi?

Power BI’dagi vizualizatsiyalar juda tez ishlaydi

Hisob-kitoblar osonlashadi

Relationshiplar tartibli bo‘ladi

DAX formulalar aniqroq bo‘ladi

5) Sales (sotuv) ma’lumotlar to‘plamida odatda fact jadvali — bu “Sales” yoki “Sales Fact” jadvali bo‘ladi.

☒ Fact jadvali odatda quyidagicha bo‘ladi:

★ Sales (FactSales)

Bu jadvalda raqamli natijalar va transaksiyalar bo‘ladi:

sale_id

product_id

customer_id

date_id

quantity

unit_price

total_amount

discount

revenue

■ Dimension jadvallari esa odatda quyidagilar:

Products

Customers

Date

Store / Region

Salesperson

📌 Xulosa:

Sales datasetida eng asosiy fact jadval — bu "Sales" jadvali.

7) Sales jadvalidagi ProductID — bu mahsulot haqidagi to'liq ma'lumot boshqa jadvalda (Products.csv) saqlanadi. Sales jadvali esa faqat mahsulotga ishora qiladi.

✖ Nima uchun foreign key? (o'zbekcha tushuntirish)

✔ 1. Sales jadvali mahsulotni o'zida saqlamaydi

Sales.csv faqat sotuvni qayd qiladi:

qancha sotilgan

qancha pul chiqqan

qaysi mijoz sotib olgan

Lekin mahsulotning:

nomi

kategoriyasi

narxi

brendi

Bu ma'lumotlar Products.csv jadvalida bo'ladi.

✔ 2. ProductID — mahsulotni aniqlash uchun ishlatiladi

Sales jadvali mahsulotning to'liq ma'lumotini takrorlamaydi, faqat:

“Bu sotuv QAYSI mahsulotga tegishli?”

degan savolga javob beruvchi ProductID ni yozadi.

✓ 3. Shuning uchun ProductID — foreign key

Products jadvalidagi Primary Key = ProductID

Sales jadvalidagi Foreign Key = ProductID

📌 Xulosa:

ProductID Sales.csv ichida foreign key, chunki u mahsulotga tegishli bo'lgan ma'lumotni boshqa jadvaldan (Products.csv) ko'rsatadi, o'zi esa faqat havola sifatida ishlaydi.

9) ★ Nima uchun Star Schema performance'ni yaxshilaydi?

☑ 1. Jadvallar soni oz — strukturasi sodda

Star schema'da:

1 ta fact jadvali

bir nechta dimension jadvallari

bo'ladi.

Murakkab joinlar yo'q → hisoblash tezlashadi.

☑ 2. Joins (bog'lanishlar) minimal bo'ladi

Star schema'da barcha relationshiplar:

Fact → Dimension (1-to-Many)

shunday bo'lgani uchun Power BI juda tez join qiladi.

Murakkab snowflake yoki normalizatsiya qilingan 10 ta jadvaldan iborat struktura bo'lmaydi →
DAX formulalar tez bajariladi.

☒ 3. Data model memory'da samarali saqlanadi

Dimension jadvallari kichik bo'ladi (unique qiymatlar).

Fact jadval eng katta bo'ladi, lekin u faqat:

ID'lar

sonlar (miqdor, narx, revenue)

ni saqlaydi.

Bu Power BI engine (VertiPaq) uchun eng ideal format →
memory kam ishlatiladi → performance oshadi.

☒ 4. Compression juda yaxshi ishlaydi

VertiPaq engine dimension jadvallarni:

group qiladi

encode qiladi

dictionary yaratadi

Bu faqat star schema bilan ideal ishlaydi.

Natija:

👉 Hisoblashlar 10 baravar tezlashishi mumkin.

☒ 5. Relationshiplar yo'nalishi aniq va to'g'ri bo'ladi

Dimension → Fact (single direction)

Bunday tartib bilan:

Filterlar to'g'ri ishlaydi


Hisob-kitob chalkashmaydi

Ambiguous relationshiplar bo'lmaydi

Bu ham tezlikni oshiradi.

📌 Xulosa (1 jumlada)

Star schema Power BI'ga minimal JOIN, maksimal compression va eng optimal data model beradi — shu tufayli performance juda kuchli bo'ladi.

11)  Circular relationship nima?

Bir jadval boshqa jadvalga, u esa yana birinchi jadvalga bog'langan holat.

Masalan:

Table A → Table B → Table C → Table A

Bu holatda Power BI hisoblashlarda qaysi yo'nalishdan filter qo'llashni bilmaydi → xato yoki sekin ishlash.

☒ Circular relationships'ni yechish usullari

1 Relationship yo'nalishini tekshirish va Single direction qilish

Model View'da relationshiplarni tekshiring.

Kerak bo'lsa, Cross filter direction = Single qilib o'zgartiring.

Ko'p filter yo'nalishlari (Both) doira hosil qilishi mumkin.

2 Bridge/junction table yaratish

Agar Many-to-Many relationship doira hosil qilsa:

Ularni bog'laydigan alohida bridge table yarating.

Bridge table orqali barcha relationshiplarni single direction bilan ulang.

Misol:

Students ↔ Courses (Many-to-Many)

StudentCourses bridge table yaratish

3 Keraksiz relationshiplarni o'chirish

Ba'zan doira faqat keraksiz relationship tufayli yuzaga keladi.

Faqat eng muhim relationshipni qoldiring, boshqasini o'chiring.

4 Fact jadvalga barcha filterlarni to'plang

Star schema qurish orqali:

Barcha dimension → fact relationship single direction

Bu doira yuzaga kelmaydi

5 DAX orqali filterni boshqarish

Agar doira mavjud bo'lsa, USERRELATIONSHIP() yoki TREATAS() bilan muntazam relationshipni tanlab ishlatish mumkin.

✚ Xulosa

Circular relationship'ni hal qilishning eng oson yo'li:

Keraksiz relationshiplarni o'chirish

Cross filter direction = Single qilish

Zarur bo'lsa bridge table orqali bog'lash

14) Power BI'da bidirectional (ikki yo'nalishli) filtering kuchli vosita, lekin uni har doim ishlatish tavsiya etilmaydi, chunki bu performance pasayishi va circular relationship xatolariga olib kelishi mumkin.

Quyida o'zbekcha tushuntirib beraman.

◆ Bidirectional filtering nima?

Oddiy relationship: Single → filter faqat bir yo'nalishda ishlaydi (masalan, Dimension → Fact)

Bidirectional: Both → filter fact → dimension va dimension → fact yo'nalishida ishlaydi

◆ Qachon bidirectional filtering ishlatish mumkin?

Ko'p-to-ko'p (Many-to-Many) holatlarda

Masalan, Customers ↔ Products

Bridge table orqali bog'langan bo'lsa, Both filter bilan hisoblashing mumkin

Sanoqli va kichik modellarda

Faqat bir necha jadval bo'lsa, performance katta pasaymaydi

Filtrlar bir necha fact jadvalga tarqatilishi kerak bo'lsa

Masalan, bitta dimension jadvali bir nechta fact jadvalni filter qiladi

- ◆ Qachon ishlatmaslik kerak

Katta datasetlarda → sekin ishlaydi

Star schema'da fact → dimension filterini Both qilmaslik tavsiya etiladi

Circular relationship hosil bo'lishi mumkin

- ◆ Xulosa

Bidirectional filtering faqat zarur bo'lganda ishlatiladi:

Ko'p-to-ko'p relationship

Bir nechta fact jadvalni bitta dimension orqali filterlash

Aks holda, Single direction ishlatish eng optimal variant.

15) Power BI'da DAX orqali referential integrityni "enforce" qilish, ya'ni CustomerID o'chirib yuborilsa, Sales jadvalidagi tegishli yozuvlarni ham hisoblashdan chiqarish odatda CALCULATE + RELATEDTABLE / FILTER orqali qilinadi. Power BI'da SQL kabi avtomatik cascade delete yo'q, shuning uchun hisoblashlarda integrity'ni saqlash kerak.

- ◆ Masalan: Total Sales faqat mavjud mijozlar uchun

TotalSales_ValidCustomers =

```
CALCULATE(  
    SUM(Sales[TotalSales]),  
    FILTER(  
        Sales,  
        NOT(ISBLANK(RELATED(Customers[CustomerID])))  
    )  
)
```

- ◆ Nima qilayapti:

RELATED(Customers[CustomerID])

Har bir Sales satri uchun tegishli CustomerID ni tekshiradi

ISBLANK()

Agar Customer jadvalidagi yozuv o'chirilgan bo'lsa → BLANK() qaytaradi

NOT(ISBLANK(...))

Faqat mavjud mijozlar uchun hisoblash

CALCULATE(SUM(...), FILTER(...))

TotalSales ni faqat mavjud mijozlarga cheklaydi

◆ Xulosa

Power BI'da referential integrity "avtomatik" emas.

Mijoz o'chirilsa, tegishli Sales yozuvlari DAX formulasida filtr orqali hisoblanmaydi.

Bu usul cascade delete o'rnini bosadi va hisobotni to'g'ri qiladi.