



SQL

Ռեկացիոն բազայի կառուցվածքը

- ❑ **Առյուսակներ** – իրենից ներկայացնում է որևէ եռթյան կամ դրա մի մասին արտապատկերում տվյալների բազայում (օր. Customer, Product, Car, Person...)
- ❑ **Սյուներ** – յուրաքանչյուր սյուն իրենից ներկայացնում է առյուսակում արտապատկերված եռթյան որևէ դաշտ (օր. Name, Phone, Price, ...): Սյուները խիստ տիպավորված են:
- ❑ **Տողեր (գրառումներ)** – յուրաքանչյուր տող իրենից ներկայացնում է մեկ գրառում:
- ❑ **Բանալիներ**
 - ❑ Առաջնային բանալի – օգտագործվում է կոնկրետ գրառումը միանշանակորեն իդենտիֆիկացնելու համար: Ունիկալ է յուրաքանչյուր գրառման համար և չի կրկնվում՝ նույնիսկ գրառումը բազայից հեռացնելու դեպքում:
 - ❑ Արտաքին բանալի – կիրառվում է առյուսակները միմյանց միջև կապելու համար:



SQL Server հիմնական տիպերը

- [bigint](#)
 - [binary](#)
 - [bit](#)
 - [char](#)
 - [date](#)
 - [datetime](#)
 - [decimal](#)
 - [float](#)
 - [image](#)
 - [int](#)
 - [money](#)
 - [nchar](#)
 - [ntext](#)
 - [numeric](#)
 - [nvarchar](#)
 - [real](#)
 - [text](#)
 - [time](#)
 - [varchar](#)
-



SQL լեզվի հիմնական հրամանները

- SELECT
- INSERT
- UPDATE
- DELETE



SELECT հրամանը

- Օգտագործվում է աղյուսակից (կամ աղյուսակներից) տվյալներ ստանալու համար:
- Պարզագույն կառուցվածքը.
SELECT col1, col2, ..., col3 **FROM** tableName
- Բոլոր սյունակների ստացման համար օգտագործվում է * նշանը.
SELECT * FROM tableName (կարդացվում է select all from tablename)



SELECT հրամանը պայմանով

- **SELECT** հրամանով ստացվող արժեքները ֆիլտրելու համար օգտագործվում է **WHERE** պայմանը հետևյալ սինտաքսիսով.
SELECT col1, col2, ..,coln **FROM** tableName **WHERE**
պայմաններ
- *պայմաններ* – բլոկը պարունակում է համեմատության օպերատորներ, բուլյան օպերատորներ, ինչպես նաև կարող է պարունակել հատուկ սիմվոլներ և ֆունկցիաներ:



Չափերտների օգտագործվումը SQL հարցումներում

- Բոլոր տեքստային տիպերի սյունակների արժեքներին դիմելիս անհրաժեշտ է արժեքը ներառել չափերտների մեջ (չափերտների տիպը (միակի կամ կրկնակի) կախված է ՏԲՂ-ից, ինչպես նաև՝ ծրագրավորման լեզվի առանձնահատկություններից):
օր. Ստանալ բոլոր այն աշխատողների ազգանունները, որոնց անունը Nancy է.

```
SELECT LastName FROM Employees WHERE  
(FirstName = 'Nancy')
```

- Թվային տիպերին դիմելիս չափերտներ չեն օգտագործվում.
օր. Ստանալ բոլոր այն ապրանքների անվանումները, որոնց գինը մեծ է 30-ից

```
SELECT ProductName FROM Products WHERE  
(UnitPrice > 30)
```



Դուբլիկատների վերացումը **SELECT** հարցման մեջ

□ **SELECT** հարցումը վերադարձնում է բոլոր այն տողերը, որոնք համապատասխանում են **WHERE** պայմանին (դրա առկայության դեպքում)՝ առանց դուբլիկատների վերացման:

□ Եթե անհրաժեշտ է ստանալ որևէ սյան (սյուների) ունիկալ արժեքները, ապա անհրաժեշտ է օգտագործել **DISTINCT** օպերատորը՝ **SELECT** հարցման մեջ.

SELECT DISTINCT colName **FROM** tableName

□ Եթե **DISTINCT** օպերատորի հետ օգտագործվում են մի քանի սյունակներ, ապա ունիկալությունը ապահովվում է այդ այդ սյունակների համադրությամբ՝ այսինքն բացառվում է դրանց համատեղ կրկնությունը:

Հարցման արդյունքների սորտավորումը

- ORDER BY օպերատորի կիրառումը SELECT հարցման մեջ թույլ է տալիս հարցման արդյունքները ստանալ դասավորված աճման կամ նվազման կարգով՝ ըստ որևէ սյունյակի (սյունյակների) արժեքների:
 - **SELECT * FROM** tableName **Order By** column:
 - Լռելյայն դասավորումը կատարվում է աճման կարգով: Այդ պահվածքը կարելի է սահմանել Asc կամ Desc օպերատորների կիրառմամբ.
SELECT * FROM tableName **Order By** column **Desc**:
 - **Order By** օպերատորի հետ մի քանի սյունյակների օգտագործման դեպքում սորտավորումը կատարվում է ըստ նշված սյունյակների՝ ձախից աջ հերթականությամբ:
-



WHERE օպերատորի կիրառությունը բազմության հետ:

- Եթե անհրաժեշտ է նշալ պայմանի պատկանելիությունը որևէ դիապազոնի, ապա կարելի է օգտագործել BETWEEN հրամանը.
օր.

```
SELECT * FROM Customers where city Between 'a'  
and 'd'
```

- Ստուգելու համար պայմանի պատկանելիությունը որոշակի բազմության, օգտագործվում է IN հրամանը.
... where colName in (val1, val2, ...,valN)



Նշված շաբլոնին համապատասխանության ստուգումը: Like

- LIKE օպերատորի միջոցով կարելի է սահմանել տողի շաբլոն (frame): Այն աշխատում է երկու հատուկ սիմվոլների հետ.
 - % - արտահայտում է կամայական քանակի սիմվոլներ:
 - — (underline) – արտահայտում է մեկ կոնկրետ սիմվոլ:
- Օր. ... where sity like ‘a%’ – քաղաքի անվանումը սկսվում է a տառով և ունի կամայական երկարություն:
... where name like ‘go_’ անունը սկսվում է go տառերով և ունի ճիշտ 3 սիմվոլ երկարություն:
SELECT * FROM Customers where city like ‘%a_’ – քաղաքի անվանման նախավերջին տառը a է:



ESCAPE հրամանի օգտագործումը

- Եթե անհրաժեշտ է LIKE հրամանի միջոցով ստանալ այնպիսի տողեր, որոնք պարունակում են ‘%’ կամ ‘_’ ղեկավարող սիմվոլները, ապա անհրաժեշտ է տեղեկացնել ղեկավարող համակարգին, որ տվյալ սիմվոլը (% , _) ոչ թե հատուկ հրաման է, այլ սովորական սիմվոլ:
 - Այդ նպատակով օգտագործվում են ESCAPE հրամանը: Այն թույլ է տալիս սահմանել այնպիսի սիմվոլ, որի օգտագործումը ղեկավարող սիմվոլից առաջ վերածում է այն տառացի սիմվոլի:
օր. **SELECT * FROM Customers where city like '%a_'**
ESCAPE '\' – քաղաքի անվանումը ավարտվում է a_ տողով:
-



NULL տիպի ստուգումը

- Տվյալների բազայում NULL նշանակում է տվյալի բացակայություն: Այդ արժեքի պարունակումը թույլատրվում է միայն Allow Null տիպի սյունյակներում: NULL արժեքի ստուգումը հնարավոր չէ իրականացնել համեմատության օպերատորների (=) օգտագործմամբ: Այդ նպատակով օգտագործվում է IS NULL պրեդիկատները:
- Օր. ստանալ բոլոր այն աշխատողների անունները, որոնց բնակության քաղաքը նշված չէ.

```
SELECT FirstName FROM Employees where City IS  
NULL
```



NOT պրեդիկատը

- NOT-ը օգտագործվում է BETWEEN, IN, LIKE և IS NULL հրամանների հետ՝ դրանց արդյունքները ժխտելու համար:
- Օր. ստանալ այն Customer-ներին, որոնց քաղաքի անունը սկսվում է ցանկացած տառով, բացի a-ից.
SELECT * FROM Customers where City not like 'a%'
- **SELECT** FirstName **FROM** Employees **where** City IS NOT NULL



Տվյալների ներմուծում բազա: INSERT հրամանը

- **INSERT** tableName (col1, col2, ..., colN) **VALUES** (val1, val2, ..., valN)
- (col1, col2, ..., colN) – մասը կարելի է բաց թողնել: Այդ դեպքում (val1, val2, ..., valN) արժեքները պետք է ճշգրտորեն համապատասխանեն տվյալ աղյուսակում առկա սյուների քանակին և հերթականությանը:
- ՉԳՈՒՇԱՑՈՒՄ Unicode ինֆորմացիան կորեկտ ներմուծելու համար անհրաժեշտ է օգտագործել N սիմվոլը ուղարկվող տողից առաջ (SQL Server):

օր. insert into TestTable (Name) values(N'Բարև')



Տվյալների խմբագրումը: Update

- Ընհանուր տեսքն է.

UPDATE tableName SET col1 = val1, ..., colN = valN
WHERE condition:

- ՉԳՈՒՇԱՑՈՒՄ. WHERE հատվածի բացթողման դեպքում կտարմացվեն տվյալ աղյուսակի բոլոր տողերը:



Տվյալների հեռացումը բազայից. DELETE

- Ընդհանուր տեսքը.

DELETE FROM tableName WHERE condition



Մի քանի աղյուսակների միավորումը

Աղյուսակների միավորման տիպերը

- Մեկը մեկին
- Մեկը շատին
- Շատը շատին



Միավորված աղյուսակներից տվյալների ստացումը: JOIN-հրամաններ

- JOIN – հրամանները թույլ են տալիս ստանալ երկու աղյուսակների միավորումը: Ընդհանուր տեսքն է.

SELECT * **from** TableA xxx **join** TableB **on** TableA.Col
համեմատության օպերատոր **TableB.Col**

- Գոյություն ունի Join հրամանի մի քանի տարատեսակներ: Ամենակիրառականն են.
 - INNER JOIN (CROSS JOIN, JOIN)
 - LEFT JOIN
 - RIGHT JOIN
 - FULL OUTER JOIN



INNER JOIN

- ❑ INNER JOIN – վերադարձնում է միայն այն տողերը, որոնք միաժամանակ գոյություն ունեն և առաջին և երկրորդ աղյուսակներում.
օր.
- ❑ `SELECT Products.ProductName, Categories.CategoryName from Products inner join Categories on Products.CategoryID = Categories.CategoryID`
- ❑ Այս դեպքում կստացվեն միայն այն Product-ները, որոնց Category-ն նշված է
- ❑ INNER բառը կարելի է բաց թողնել



LEFT JOIN

- LEFT JOIN հրամանի դեպքում ձախ աղյուսակը համարվում է ղեկավարող: Դրանից վերցվում են բոլոր արժեքները, այնուհետև կատարվում է փնտրում աջ աղյուսակում: Բոլոր այն արժեքները, որոնք բացակայում են աջ աղյուսակում, փոխարինվում են null-ով:
- Օր. `SELECT Products.ProductName, Categories.CategoryName from Products left join Categories on Products.CategoryID = Categories.CategoryID`
- Այս դեպքում կստացվեն բոլոր Product-ները, իսկ նրանց Category-ն, որոնցը նշված չէ, կլինի null



RIGHT JOIN

- RIGHT JOIN-ի դեպքում ղեկավարող համարվում է աջ աղյուսակը:



FULL OUTER JOIN

- Վերադարձնում է բոլոր տողերը և՛ ձախ և՛ աջ սյունակներում: Բացակայող արժեքները փոխարինվում են null-ով:

