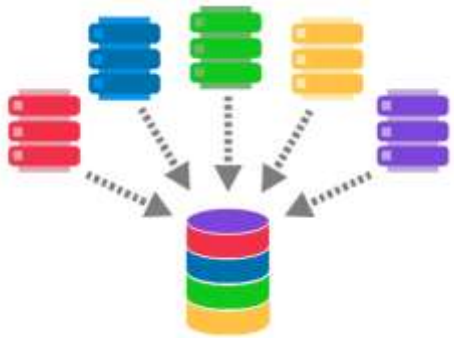
Baze podataka II
Modul 3 – Jezik SQL

Uvod u SQL/DML

Grupisanje i sumiranje podataka



Summary

- Upotreba agregatnih funkcija
- Sumiranje grupisanih podataka
- Logičko procesiranje upita

Lekcija 1: Upotreba agregatnih funkcija

- Pregled funkcija za agregacije
- Funkcije za agregacije i NULL
- Agregatne funkcije i uslovi

Pregled funkcija za agregacije

- **Funkcije za agregacije se koriste za kalkulacije nad skupom vrijednosti gdje je rezultat jedna vrijednost**
- **Uglavnom se koriste za sumiranje podataka u kolonama:**
  - Prosjek ocjena;
  - Ukupna prodaja
  - i sl.
- **Osnovne funkcije su: (MIN, MAX, AVG, SUM i COUNT)**

```
SELECT MAX (ListPrice) AS MaxListPrice,
       MIN (ListPrice) AS MinListPrice,
   AVG (CONVERT (int, Size)) AS AvgProductSize,
   SUM (DaysToManufacture) AS TotalDaysToManifacture
FROM Production.Product
WHERE ISNUMERIC (Size) = 1
```

Funkcije za agregacije i NULL

- **Većina agregatnih funkcija ignoriše NULL**
  - Mogu se pojaviti anomalije u rezultatima
  - Kako da nešto brojimo, sumiramo što ima kategoriju NULL
- **Funkcija ISNULL može korigovati ove slučajeve**

```
SELECT COUNT (*), COUNT (Weight),
   AVG (Weight), AVG (ISNULL (Weight, 0))
FROM Production.Product
```

- **Funkcija COUNT (*) ignoriše NULL vrijednost**

Agregatne funkcije i uslovi

- **WHERE se može koristiti kao klasično filtriranje pretrage**

```
SELECT SUM (OrderQty) AS TotalOrderQty
FROM Purchasing.PurchaseOrderDetail
WHERE ProductID = 512 AND DueDate = '2005-06-14'
```

- **Ali postoji ograničenje ako pokušamo koristiti funkciju za agregaciju u kombinaciji sa nekim od operatora (izraz)**

```
SELECT SUM (OrderQty) AS TotalOrderQty
FROM Purchasing.PurchaseOrderDetail
WHERE SUM (OrderQty) > 1000
```

Lekcija 2: Sumiranje grupisanih podataka

- GROUP BY klauzula
- HAVING klauzula

GROUP BY klauzula...

- **Pravi sumarnu vrijednost za agregatne funkcije**
- **Mandatorna ako u SELECT listi želimo koristiti kolone, a da iste nisu unutar funkcije za agregaciju**

```
SELECT ProductID, COUNT (ProductID) AS ProductSales,
  SUM (LineTotal) As Profit
FROM Purchasing.PurchaseOrderDetail
GROUP BY ProductID
```

- **Prilikom rada sa GROUP BY potrebno ja upamtiti:**
  - Sve kolone koje su u SELECT listi, a nisu pod agregacijom moraju biti u GROUP BY
  - NULL vrijednosti se tretiraju jednako kao i sve ostale tj. grupišu se

...GROUP BY klauzula

```
SELECT ProductID, OrderID ,Quantity
FROM Orders
```

ProductID	OrderID	Quantity
1	1	5
1	1	10
2	1	10
2	2	25
3	1	15
3	2	30

ProductID	Quantity
1	15
2	35
3	45

```
SELECT ProductID ,
       SUM(Quantity) AS TotalQuantity
FROM Orders
GROUP BY ProductID
```

HAVING klauzula

- **HAVING se ponaša slično kao i WHERE i koristi se za filtriranje rezultata pretrage**
- **Zna da rukuje sa grupama koje vrati klauzulu GROUP BY**

```
SELECT SUM (OrderQty) AS TotalOrderQty
FROM Purchasing.PurchaseOrderDetail
HAVING SUM (OrderQty) > 1000
```

- **Ako je agregacija uslov filtriranja i/ili dio izraza onda je HAVING mandatoran**
- **Aliasi se ne mogu koristiti u HAVING**
  - procesira se prije SELECT liste

Lekcija 3: Logičko procesiranje upita

- Šta je logičko procesiranje?
- Faze logičkog procesiranja;
- Primjer scenarija;

Šta je logičko procesiranje?

- Logičko procesiranje se sastoji od koraka koji se izvode prema određenom redoslijedu bez mogućosti;
- Možemo reći da logičko procesiranje vodi prema izlazu upita tj. njegovom rezultatu
- Pisanje upita bez poznavanja navedenog vodi kao lošim upitima i kodu koji je teško održavati;

Faze logičkog procesiranja

- Faze (korake) logičkog procesiranja možemo objasniti na primjeru opšte sintakse SELECT iskaza;

```
(8)   SELECT (9) DISTINCT (11)TOP
(1)   FROM <lijeva_tabela>
(3)     <tip_spoja> JOIN <desna_tabela>
(2)       ON <uslov_spajanja>
(4)   WHERE
(5)   GROUP BY
(6)   WITH {CUBE | ROLLUP}
(7)   HAVING
(10)  ORDER BY
```

Primjer scenarija

Kupci	
KupacID	**Grad**
Sara	Mostar
Jasmin	Mostar
Imran	Mostar
Selver	Vareš

Narudzbe	
NarudzbaID	KupacID
1	Jasmin
2	Jasmin
3	Imran
4	Imran
5	Imran
6	Selver
7	NULL

```
SELECT K.KupacID, COUNT(N.NarudzbaID) AS BrojNarudzbi
FROM Kupci AS K
  LEFT OUTER JOIN Narudzbe AS N
    ON K.KupacID = N.KupacID
WHERE K.Grad = 'Mostar'
GROUP BY K.KupacID
HAVING COUNT(N.NarudzbaID) < 3
ORDER BY BrojNarudzbi;
```

Korak 1: (Kartezijev proizvod) CROSS JOIN

K.KupacID	K.Grad	N.NarudzbaID	N.KupacID
SARA	Mostar	1	JASMIN
SARA	Mostar	2	JASMIN
SARA	Mostar	3	IMRAN
SARA	Mostar	4	IMRAN
SARA	Mostar	5	IMRAN
SARA	Mostar	6	SELVER
SARA	Mostar	7	NULL
JASMIN	Mostar	1	JASMIN
JASMIN	Mostar	2	JASMIN
JASMIN	Mostar	3	IMRAN
JASMIN	Mostar	4	IMRAN
JASMIN	Mostar	5	IMRAN
JASMIN	Mostar	6	SELVER
JASMIN	Mostar	7	NULL
IMRAN	Mostar	1	JASMIN
IMRAN	Mostar	2	JASMIN
IMRAN	Mostar	3	IMRAN
IMRAN	Mostar	4	IMRAN
IMRAN	Mostar	5	IMRAN
IMRAN	Mostar	6	SELVER
IMRAN	Mostar	7	NULL
SELVER	Vareš	1	JASMIN
SELVER	Vareš	2	JASMIN
SELVER	Vareš	3	IMRAN
SELVER	Vareš	4	IMRAN
SELVER	Vareš	5	IMRAN
SELVER	Vareš	6	SELVER
SELVER	Vareš	7	NULL

- **FROM: Kartezijev prozovod (cross join) između prve dvije tabale u FROM klauzili. Rezultat je virtulena tabela VT1**

```
FROM Kupci AS K ... JOIN Narudzbe AS N
```

Korak 2: Primjena ON filtera

Uslov?	K.KupacID	K. Grad	N.narudzbaID	N.KupacID
FALSE	SARA	Mostar	1	JASMIN
FALSE	SARA	Mostar	2	JASMIN
FALSE	SARA	Mostar	3	IMRAN
FALSE	SARA	Mostar	4	IMRAN
FALSE	SARA	Mostar	5	IMRAN
FALSE	SARA	Mostar	6	SELVER
UNKNOWN	SARA	Mostar	7	NULL
TRUE	JASMIN	Mostar	1	JASMIN
TRUE	JASMIN	Mostar	2	JASMIN
FALSE	JASMIN	Mostar	3	IMRAN
FALSE	JASMIN	Mostar	4	IMRAN
FALSE	JASMIN	Mostar	5	IMRAN
FALSE	JASMIN	Mostar	6	SELVER
UNKNOWN	JASMIN	Mostar	7	NULL
FALSE	IMRAN	Mostar	1	JASMIN
FALSE	IMRAN	Mostar	2	JASMIN
TRUE	IMRAN	Mostar	3	IMRAN
TRUE	IMRAN	Mostar	4	IMRAN
TRUE	IMRAN	Mostar	5	IMRAN
FALSE	IMRAN	Mostar	6	SELVER
UNKNOWN	IMRAN	Mostar	7	NULL
FALSE	SELVER	Vareš	1	JASMIN
FALSE	SELVER	Vareš	2	JASMIN
FALSE	SELVER	Vareš	3	IMRAN
FALSE	SELVER	Vareš	4	IMRAN
FALSE	SELVER	Vareš	5	IMRAN
TRUE	SELVER	Vareš	6	SELVER
UNKNOWN	SELVER	Vareš	7	NULL

- **ON: Primjenjuje se ON filter na VT1: Podaci iz tabela gdje je uslov spajanja TRUE se ubacuju u VT2**

```
ON K.KupacID = N.KupacID
```

Uslov?	K.KupacID	K. Grad	N.NarudzbaID	N.KupacID
TRUE	JASMIN	Mostar	1	JASMIN
TRUE	JASMIN	Mostar	2	JASMIN
TRUE	IMRAN	Mostar	3	IMRAN
TRUE	IMRAN	Mostar	4	IMRAN
TRUE	IMRAN	Mostar	5	IMRAN
TRUE	SELVER	Vareš	6	SELVER

Korak 3: Dodavanje vanjskih (OUTER) redova

- **OUTER JOIN je suprotan od CROSS JOIN i redovi iz „preserved” tabele ili oni koji nemaju TRUE iz stavke 2 se spajaju sa VT2 i kreiraju VT3**

K.KupacID	K. Grad	N.NarudzbaID	N.KupacID
JASMIN	Mostar	1	JASMIN
JASMIN	Mostar	2	JASMIN
IMRAN	Mostar	3	IMRAN
IMRAN	Mostar	4	IMRAN
IMRAN	Mostar	5	IMRAN
SELVER	Vareš	6	SELVER
SARA	Mostar	NULL	NULL

```
Kupci AS K LEFT OUTER JOIN Narudzbe AS N
```

Korak 4: Primjena (WHERE) filtera

- **Drugi od tri filtera iz upita eliminiše sve zapise iz VT3 koji nemaju vrijednost TRUE za zadati uslov. Kao što vidite izbačen je jedan zapis i kreira se VT4.**

K.KupacID	K. Grad	N.NarudzbaID	N.KupacID
JASMIN	Mostar	1	JASMIN
JASMIN	Mostar	2	JASMIN
IMRAN	Mostar	3	IMRAN
IMRAN	Mostar	4	IMRAN
IMRAN	Mostar	5	IMRAN
SARA	Mostar	NULL	NULL

```
WHERE K.Grad = 'Mostar'
```

Korak 5: Grupisanje (GROUP BY)

- **GROUP BY: Redovi iz VT4 se grupišu prema uslovu iz klauzulu GROUP BY, nastaje VT5**

Grupa				
K.KupacID	**K. Grad**	**N.NarudzbaID**	**N.KupacID**	**K.KupacID**
JASMIN	JASMIN	Mostar	1	JASMIN
	JASMIN	Mostar	2	JASMIN
IMRAN	IMRAN	Mostar	3	IMRAN
	IMRAN	Mostar	4	IMRAN
	IMRAN	Mostar	5	IMRAN
SARA	SARA	Mostar	NULL	NULL

```
GROUP BY K.KupacID
```

Korak 6: Primjena HAVING filtera

- **HAVING: filter se primjenjuje i vraćaju samo oni zapisi sa TRUE elementom VT6**

K.KupacID	K. Grad	N.NarudzbaID	N.KupacID	K.KupacID
JASMIN	JASMIN	Mostar	1	JASMIN
	JASMIN	Mostar	2	JASMIN
SARA	SARA	Mostar	NULL	NULL

```
HAVING COUNT(N.NarudzbaID) < 3
```

Korak 7: Procesiranje SELECT liste

- Na kraju smo dobili finalni oblik ili VT7

K.KupacID	BrojNarudzbi
JASMIN	2
SARA	0

```
SELECT K.KupacID, COUNT(N.NarudzbaID) AS BrojNarudzbi
```

Pitanja