Database Management Systems

SQL Query Language (3)

Topics

- Aggregate Functions in Queries
  - count
  - sum
  - max
  - min
  - avg
- Group by queries
- Set Operations in SQL Queries
- Views

Aggregate Functions

- Tables are collections of records with a schema.
- Aggregate functions are used to get information about the records of a table such as:
  - The maximum value in a column of a table
  - The number of records in a table
  - Average of the values in a column of a table

Aggregate Functions

- Most important aggregate functions are:
  - count
  - sum
  - max
  - min
  - avg

Count Function

- **Count** returns the number of rows or distinct values.
- **Count** is used with **SELECT** query
- **Count** can be used with **distinct** option

- Syntax

  Select count([distinct] attribute list)

  From table_name

  where condition

Example

- Find the number of employees:

  Select count(*)

  From Employee

- Find the number of different values on the attribute Salary for all the rows in Employee:

  Select count(distinct Salary)

  From Employee

Example Query Results

EMPLOYEE

FirstName	Surname	Dept	Office	Salary	City
Mary	Brown	Administration	10	45	London
Charles	White	Production	20	36	Toulouse
Gus	Green	Administration	20	40	Oxford
Jackson	Neri	Distribution	16	45	Dover
Charles	Brown	Planning	14	80	London
Laurence	Chen	Planning	7	73	Worthing
Pauline	Bradshaw	Administration	75	40	Brighton
Alice	Jackson	Production	20	46	Toulouse

- Query 1 : 8
- Query 2 : 6

Sum Function

- Sum finds the total sum of the values in an attribute.
- Sum cannot be used with non-numeric attributes.

e.g. Find the sum of the salaries of the Administration department

```
select sum(Salary) as SumSalary
from Employee
where Dept = 'Administration'
```

Result:

SumSalary
125

Min, Max and Avg functions

- Min, Max, and Avg functions are used to find the minimum, maximum and average of the values in an attribute.

e.g. Find the maximum salary among the employees who work in a department based in London.

Example Query

```
Select max(Salary) as MaxLondonSal
From Employee, Department
Where
    Employee.DeptCode = Department.DeptCode
    and
    Department.City = 'London'
```

Result:

MaxLondonSal
80

Group By

- Aggregate functions can also be used with subsets of the table records.
- **Group By** puts the records in subsets (groups)
- Aggregate functions can be used with **Group By**

EMPLOYEE

FirstName	Surname	DeptCode	Office	Salary	City
Mary	Brown	1001	10	45	London
Charles	White	1002	20	36	Toulouse
Gus	Green	1001	20	40	Oxford
Jackson	Neri	1003	16	45	Dover
Charles	Brown	1004	14	80	London
Laurence	Chen	1004	7	73	Worthing
Pauline	Bradshaw	1001	75	40	Brighton
Alice	Jackson	1002	20	46	Toulouse

DEPARTMENT

DeptCode	DeptName	Address	City
1001	Administration	Bond Street	London
1002	Production	Rue Victor Hugo	Toulouse
1003	Distribution	Pond Road	Brighton
1004	Planning	Bond Street	London
1005	Research	Sunset Street	San José

Group By Example

- Find the sum of salaries of all the employees of each department:

```
Select Dept, sum(Salary)as TotSal
From Employee
Group by Dept
```

Result:

Dept	TotSal
Administration	125
Distribution	45
Planning	153
Production	82

How Group By Works?

First the query is executed

Then the query result is divided into subsets

Next aggregate function is executed for each group

Dept	Salary
Administration	45
Administration	40
Administration	40
Distribution	45
Planning	80
Planning	73
Production	36
Production	46

Dept	TotSal
Administration	125
Distribution	45
Planning	153
Production	82

Conditions on Aggregate Function Result

- When conditions are on the result of an aggregate operator, it is necessary to use the **having** clause

- e.g. Find which departments spend more than 100 on salaries

```
Select Dept, sum(Salary) as TotSal
from Employee
group by Dept
having sum(Salary) > 100
```

Result:

Dept	TotSal
Administration	125
Planning	153

Example

- Find the departments in which the average salary of employees older than 30 is higher than 25:

```
Select Dept, avg(Salary)
from Employee
where Age > 30
group by Dept
having avg(Salary) > 25
```

Set Operations in SQL Queries

- Union, Intersection and Set Difference operations can be carried out with SQL
- e.g. Find the first names and surnames of the employees

This query tries to find the union of two sets

1- Set of first names

2- Set of surnames

Union in SQL Queries

```
Select FirstName as Name
from Employee
union
Select Surname
from Employee
```

- Duplicates are removed by UNION

Union Query Result

Name
Mary
Charles
Gus
Jackson
Laurence
Pauline
Alice
White
Green
Neri
Brown
Chen
Bradshaw

FirstName	Surname
Mary	Brown
Charles	White
Gus	Green
Jackson	Neri
Charles	Brown
Laurence	Chen
Pauline	Bradshaw
Alice	Jackson

Intersection in SQL

Find the surnames of employees that are also first names:

```
select FirstName as Name
from Employee
intersect
select Surname
from Employee
```

Intersect Query Result

Name
Jackson

FirstName	Surname
Mary	Brown
Charles	White
Gus	Green
Jackson	Neri
Charles	Brown
Laurence	Chen
Pauline	Bradshaw
Alice	Jackson

Set Difference in SQL

- Find the first names of employees that are not also surnames:

```
select FirstName as Name
from Employee
except
select Surname
from Employee
```

Set Difference Result

Name
Mary
Charles
Gus
Charles
Laurence
Pauline
Alice

FirstName	Surname
Mary	Brown
Charles	White
Gus	Green
Jackson	Neri
Charles	Brown
Laurence	Chen
Pauline	Bradshaw
Alice	Jackson

Views

- A view is a virtual table. It does not physically exist. Rather, it is created by a query joining one or more tables
- A view can be a subset of a table
- Syntax:

  create view *ViewName [(AttributeList)] as SelectSQL*

Creating a Sample View

- Create a view for the employees of Administration department using their Name, Surname, and Salary.

- The view is the result of a select query on the employee table

Example Creating Views

```
Create view AdminEmployee
(FirstName,Surname,Salary) as

select FirstName,Surname,Salary
 from Employee
 where Dept = 'Administration'
```

Example View

EMPLOYEE

FirstName	Surname	Dept	Office	Salary	City
Mary	Brown	Administration	10	45	London
Charles	White	Production	20	36	Toulouse
Gus	Green	Administration	20	40	Oxford
Jackson	Neri	Distribution	16	45	Dover
Charles	Brown	Planning	14	80	London
Laurence	Chen	Planning	7	73	Worthing
Pauline	Bradshaw	Administration	75	40	Brighton
Alice	Jackson	Production	20	46	Toulouse

ADMINEMPLOYEE

FirstName	Surname	Salary
Mary	Brown	45
Gus	Green	40
Pauline	Bradshaw	40

Summary

- Aggregate functions are used to get information about the records of a table
- Set operations are also used in SQL tables
- A virtual table named view can be created using some of the records or attributes of one or more tables.

Questions?