


SQL language: basics

0

SQL language: basics


- SQL Language
- Language Instruction
- Sample notation and database
- SELECT Statement
- Aggregate Functions
- Operator GROUP BY



1

The SQL language


- A language for managing relational databases
 - Structured Query Language
- SQL provides commands to
 - define the schema of a relational database
 - read and write data
 - define the schema of derived tables
 - define user access privileges
 - manage transactions
- The SQL language may be used in two ways
 - interactive
 - compiled
 - a host language encapsulates the SQL commands
 - SQL commands can be distinguished from the host language commands by means of appropriate syntactic mechanisms



2

The SQL language


- SQL is a *set-level* language
 - operators are applied to relations (tables)
 - the result is always a relation (table)
- SQL is a *declarative* language
 - it describes *what to do* and not how to do it
 - it has a higher level of abstraction compared to traditional programming languages



3

SQL instructions


The SQL language



4

The SQL language

- Can be divided into
 - DML (Data Manipulation Language)
 - language for querying and updating the data
 - DDL (Data Definition Language)
 - language for defining the database structure



5

Data Manipulation Language

- To query a database in order to extract data of interest
 - SELECT
- To modify a database instance
 - INSERT: insertion of new information into a table
 - UPDATE: update of the information in the database
 - DELETE: cancellazione di dati obsoleti

DBG 6

Data Definition Language

- To define a database schema
 - creation, modification and deletion of tables: CREATE, ALTER, DROP TABLE
- To define derived tables
 - creation, modification and deletion of tables whose content is obtained from other database tables: CREATE, ALTER, DROP VIEW
- To define complementary data structures for efficiently retrieving the data
 - creation and deletion of indices: CREATE, DROP INDEX
- To define user access privileges
 - grant and revocation of privileges on resources: GRANT, REVOKE
- To define transactions
 - termination of a transaction: COMMIT, ROLLBACK

DBG 7

6

7

Notation and example database

The SQL Language

DBG 8

Syntax of SQL commands

- Notation
 - language keywords
 - upper case
 - variable terms
- Grammar
 - angle brackets <>
 - to isolate a syntactic term
 - square brackets []
 - the enclosed term is optional
 - braces {}
 - the enclosed term may not appear or maybe repeated an arbitrary number of items
 - vertical bar |
 - a term must be chosen among the options separated by the vertical bars

DBG 9

8

9

Example database: Supply-Product

PId	PName	Color	Size	Store
P1	Jumper	Red	40	London
P2	Jeans	Green	48	Paris
P3	Blouse	Blue	48	Rome
P4	Blouse	Blue	44	London
P5	Skirt	Blue	40	Paris

SId	SName	#Employees	City
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

SId	PId	Qty
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P3	200

DBG 10

10

Example database: Supply-Product


- Supplier and part DB
 - table P describes the available products
 - primary key: PId
 - table S describes the suppliers
 - primary key: SId
 - table SP describes supplies, by relating each product to the suppliers that provide it
 - primary key: (SId, PId)
 - PId: Foreign key (SP) REFERENCES PId(P)
 - SId: Foreign key (SP) REFERENCES SId(S)

DBG 11

11

The SELECT statement: basics

The SQL language




12

SELECT

```

SELECT [DISTINCT] ListOfAttributesToDisplay
FROM ListOfTablesToUse
[WHERE TupleConditions]
[GROUP BY ListOfGroupingAttributes]
[HAVING AggregateConditions]
[ORDER BY ListOfOrderingAttributes]
    
```



13

Basic SELECT(n.1)

- Find the codes and the number of employees of the suppliers based in Paris


```

SELECT SId, #Employees
FROM S
WHERE City='Paris';
    
```

SId	SName	#Employees	City
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

$$\begin{matrix}
 R \\
 \parallel \\
 \pi_{SId, \#Employees} \\
 \sigma_{City='Paris'} \\
 S
 \end{matrix}$$

SId	#Employees
S2	10
S3	30



14

Basic SELECT(n.2)

- Find the codes of all products in the database


```

SELECT PId
FROM P;
    
```

PId	PName	Color	Size	Shore
P1	Jumper	Red	40	London
P2	Jeans	Green	48	Paris
P3	Blouse	Blue	48	Rome
P4	Blouse	Blue	44	London
P5	Skirt	Blue	40	Paris
P6	Shorts	Red	42	London

$$\begin{matrix}
 R \\
 \parallel \\
 \pi_{PId} \\
 P
 \end{matrix}$$

PId
P1
P2
P3
P4
P5
P6



15

Basic SELECT(n.3)

- Find the codes of the products supplied by at least one supplier


```

SELECT PId
FROM SP;
    
```

SP	SId	PId	Qty
S1	P1	300	
S1	P2	200	
S1	P3	400	
S1	P4	200	
S1	P5	100	
S1	P6	100	
S2	P1	300	
S2	P2	400	
S3	P2	200	
S4	P3	200	
S4	P4	300	
S4	P5	400	

$$\begin{matrix}
 R \\
 \parallel \\
 \pi_{PId} \\
 SP
 \end{matrix}$$

PId
P1
P2
P3
P4
P5
P6
P1
P2
P2
P3
P4
P5



16

Basic SELECT(n.3)


- Find the codes of the products supplied by at least one supplier

```

SELECT PId
FROM SP;
    
```

It does not eliminate duplicates

$$\begin{matrix}
 R \\
 \parallel \\
 \pi_{PId} \\
 SP
 \end{matrix}$$



17

Elimination of duplicates: DISTINCT

- **DISTINCT** keyword allows the elimination of duplicates
- Find the codes of the *distinct* products supplied by at least one supplier

```
SELECT DISTINCT PId
FROM SP;
```

SP		
SId	PId	Qty
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P3	200
S4	P4	300
S4	P5	400

R
PId
P1
P2
P3
P4
P5
P6

DBG

18

18

Selection of all information

- Find **all** information related to products

```
SELECT PId, PName, Color, Size, Store
FROM P;
```

OR

```
SELECT *
FROM P;
```

R	PId	PName	Color	Size	Store
P1	Jumper	Red	40	London	
P2	Jeans	Green	48	Paris	
P3	Blouse	Blue	48	Rome	
P4	Blouse	Blue	44	London	
P5	Skirt	Blue	40	Paris	
P6	Shorts	Red	42	London	

DBG

19

19

Selection with an expression

- Find the codes of the products and the sizes expressed with the US standard

```
SELECT PId, Size-14 [AS USSize]
FROM P;
```

P					R	
PId	PName	Color	Size	Store	PId	USSize
P1	Jumper	Red	40	London	P1	26
P2	Jeans	Green	48	Paris	P2	34
P3	Blouse	Blue	48	Rome	P3	34
P4	Blouse	Blue	44	London	P4	30
P5	Skirt	Blue	40	Paris	P5	26
P6	Shorts	Red	42	London	P6	38

- Definition of a new *temporary* column for the computed expression
- the name of the temporary column may be defined by means of the **AS** keyword

DBG

20

20

The WHERE clause

- It allows expressing selection conditions applied to each tuple individually
- A Boolean expression composed by one or more predicates
- Simple predicates
 - comparison between attributes and constants
 - text search
 - NULL values

21

21

The WHERE clause (n.1)

- Find the codes of the suppliers based in Paris

```
SELECT SId
FROM S
WHERE City='Paris';
```

F				R
SId	SName	#Employees	City	SId
S1	Smith	20	London	S2
S2	Jones	10	Paris	S3
S3	Blake	30	Paris	
S4	Clark	20	London	
S5	Adams	30	Athens	

DBG

22

The WHERE clause (no.2)

- Find the codes and the number of employees of the suppliers that are not based in Paris

```
SELECT SId, #Employees
FROM S
WHERE City<>'Paris';
```

F				R	
SId	SName	#Employees	City	SId	#Employees
S1	Smith	20	London	S1	20
S2	Jones	10	Paris	S4	20
S3	Blake	30	Paris	S5	30
S4	Clark	20	London		
S5	Adams	30	Athens		

DBG

23

Boolean expressions (no.1)

- Find the codes of the suppliers based in Paris that have more than 20 employees

```
SELECT SId
FROM S
WHERE City='Paris' AND #Employees>20;
```

S				R
SId	SName	#Employees	City	SId
S1	Smith	20	London	
S2	Jones	10	Paris	
S3	Blake	30	Paris	S3
S4	Clark	20	London	
S5	Adams	30	Athens	

DBG

24

24

Boolean expressions (no.2)

- Find the codes and the number of employees of the suppliers based in Paris or London

```
SELECT SId, #Employees
FROM S
WHERE City='Paris' OR City='London';
```

S				R	
SId	SName	#Employees	City	SId	#Employees
S1	Smith	20	London	F1	20
S2	Jones	10	Paris	F2	10
S3	Blake	30	Paris	F3	30
S4	Clark	20	London	F4	20
S5	Adams	30	Athens		

DBG

25

25

Boolean expressions (no.3)

- Find the codes and the number of employees of the suppliers based in Paris and in London
 - the query may not be satisfied
 - each supplier has only one city

S			
SId	SName	#Employees	City
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

DBG

26

26

Text search

- LIKE operator**

AttributeName LIKE CharacterString

- the `_` character represents a single arbitrary character (non-empty)
- the `%` character represents an arbitrary sequence of characters (possibly empty)

27

27

Text search (no.1)

- Find the codes and the names of the products whose name begins with the letter B

```
SELECT PId, PName
FROM P
WHERE PName LIKE 'B%';
```

P					R	
PId	PName	Color	Size	Store	PId	PName
P1	Jumper	Red	40	London		
P2	Jeans	Green	48	Paris		
P3	Blouse	Blue	48	Rome	P3	Blouse
P4	Blouse	Blue	44	London	P4	Blouse
P5	Skirt	Blue	40	Paris		
P6	Shorts	Red	42	London		

DBG

28

28

Text search (no.2)

- The Address attribute contains the string 'London'
Address LIKE '%London%'

- The supplier identification number is 3 and
 - it is preceded by a single unknown character
 - it is exactly 2 characters long

SId LIKE '_3'

- The Store attribute does not have an 'e' in the second position
Store NOT LIKE '_e%'

DBG

29

29

Searching for NULL values

- IS special operator
AttributeName IS [NOT] NULL
- With NULL values, any comparison predicate is false

30

Managing NULL values

- Find the codes and the names of products with a size greater than 44

```
SELECT PId, PName
FROM P
WHERE Size>44;
```

P

PId	PName	Color	Size	Store
P1	Jumper	Red	40	London
P2	Jeans	Green	48	Paris
P3	Blouse	Blue	48	Rome
P4	Blouse	Blue	44	London
P5	Skirt	Blue	NULL	Paris
P6	Shorts	Red	42	London

R

PId	PName
P2	Jeans
P3	Blouse

- The tuples with NULL size are not selected: the predicate Size>44 evaluates to false
- With NULL values, any comparison predicate is false

DBG

31

Searching for NULL values (no.1)

- Find the codes and the names of the products whose size is unknown

```
SELECT PId, PName
FROM P
WHERE Size IS NULL;
```

P

PId	PName	Color	Size	Store
P1	Jumper	Red	40	London
P2	Jeans	Green	48	Paris
P3	Blouse	Blue	48	Rome
P4	Blouse	Blue	44	London
P5	Skirt	Blue	NULL	Paris
P6	Shorts	Red	42	London

R

PId	PName
P5	Skirt

DBG

32

32

Searching for NULL values(n.2)

- Find the codes and the names of products with a size greater than 44, or that may have a size greater than 44

```
SELECT PId, PName
FROM P
WHERE Size>44 OR Size IS NULL;
```

P

PId	PName	Color	Size	Store
P1	Jumper	Red	40	London
P2	Jeans	Green	48	Paris
P3	Blouse	Blue	48	Rome
P4	Blouse	Blue	44	London
P5	Skirt	Blue	NULL	Paris
P6	Shorts	Red	42	London

R

PId	PName
P2	Jeans
P3	Blouse
P5	Skirt

DBG

33

33

Result ordering

- ORDER BY clause
ORDER BY AttributeName [ASC | DESC]
{, AttributeName [ASC | DESC]}
- the default ordering is ascending
 - if DESC is not specified
- the ordering attributes must appear in the SELECT clause
 - even implicitly (as in SELECT *)

34

34

Result ordering (no.1)

- Find the codes of the products and their sizes, ordering the result by decreasing size

```
SELECT PId, Size
FROM P
ORDER BY Size DESC;
```

P

PId	PName	Color	Size	Store
P1	Jumper	Red	40	London
P2	Jeans	Green	48	Paris
P3	Blouse	Blue	48	Rome
P4	Blouse	Blue	44	London
P5	Skirt	Blue	40	Paris
P6	Shorts	Red	42	London

R

PId	Size
P2	48
P3	48
P4	44
P6	42
P1	40
P5	40

DBG

35

35

Result ordering (no.2)

- Find all information related to the products, ordering the result by increasing name and decreasing size

```

SELECT PId, PName, Color, Size, Store
FROM P
ORDER BY PName, Size DESC;
    
```

```

SELECT *
FROM P
ORDER BY PName, Size DESC;
    
```

R

PId	PName	Color	Size	Store
P3	Blouse	Blue	48	Rome
P4	Blouse	Red	44	London
P2	Jeans	Green	48	Paris
P1	Jumper	Red	40	London
P6	Shorts	Red	42	London
P5	Skirt	Blue	40	Paris

36

Result ordering (no.3)

- Find the codes of the products and the sizes expressed with the US standard, ordering the result by increasing size

```

SELECT PId, Size-14 AS USSize
FROM P
ORDER BY USSize;
    
```

P

PId	PName	Color	Size	Store
P1	Jumper	Red	40	London
P2	Jeans	Green	48	Paris
P3	Blouse	Blue	48	Rome
P4	Blouse	Blue	44	London
P5	Skirt	Blue	40	Paris
P6	Shorts	Red	42	London

R

PId	USSize
P5	26
P1	28
P6	28
P4	30
P2	34
P3	34

37

Join

- Defined by the **FROM** and **WHERE** clauses
- The result and efficiency of the query
 - are independent of the order of the tables in the FROM clause
 - are independent of the predicate order in the WHERE clause
 - the optimal execution order is selected by the DBMS (optimizer module)
- FROM clause with **N Tables**
 - at least **N-1** join conditions in the WHERE clause

38

Join (n.1)

- Find the names of the suppliers that provide product P2

SId	SName	#Employees	City
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

SId	PId	Qty
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200

39

Cartesian product

- Find the names of the suppliers that provide product P2

```

SELECT SName
FROM S, SP;
    
```

40

Cartesian product

S.SId	S.SName	S.#Empl	S.City	SP.SId	SP.PId	SP.Qty
S1	Smith	20	London	S1	P1	300
S1	Smith	20	London	S1	P2	200
S1	Smith	20	London	S1	P3	400
S1	Smith	20	London	S1	P4	200
S1	Smith	20	London	S1	P5	100
S1	Smith	20	London	S1	P6	100
S1	Smith	20	London	S2	P1	300
...
S2	Jones	10	Paris	S1	P1	300
...
S2	Jones	10	Paris	S2	P1	300
...

41

Join (n.1)

S.Sid	S.SName	S.#Empl	S.City	SP.Sid	SP.Pid	SP.Qty
S1	Smith	20	London	S1	P1	300
S1	Smith	20	London	S1	P2	200
S1	Smith	20	London	S1	P3	400
S1	Smith	20	London	S1	P4	200
S1	Smith	20	London	S1	P5	100
S1	Smith	20	London	S1	P6	100
S1	Smith	20	London	S2	P1	300
...
S2	Jones	10	Paris	S1	P1	300
...
S2	Jones	10	Paris	S2	P1	300
...

42

Join (n.1)

S.Sid	S.SName	S.#Empl	S.City	SP.Sid	SP.Pid	SP.Qty
S1	Smith	20	London	S1	P1	300
S1	Smith	20	London	S1	P2	200
S1	Smith	20	London	S1	P3	400
S1	Smith	20	London	S1	P4	200
S1	Smith	20	London	S1	P5	100
S1	Smith	20	London	S1	P6	100
S2	Jones	10	Paris	S2	P1	300
S2	Jones	10	Paris	S2	P2	400
S3	Blake	30	Paris	S3	P2	200
S4	Clark	20	London	S4	P3	200
S4	Clark	20	London	S4	P4	300
S4	Clark	20	London	S4	P5	400

43

Join (n.1)

- Find the names of the suppliers that provide product P2

```

SELECT SName
FROM S, SP
WHERE S.Sid=SP.Sid;
    
```

Diagram: A red arrow labeled "Join condition" points to the equals sign in the WHERE clause. Green arrows labeled "TableName.AttributeName" point from "S.Sid" and "SP.Sid" to their respective table and column names.

44

Join (n.1)

- Find the names of the suppliers that provide product P2

```

SELECT SName
FROM S, SP
WHERE S.Sid=SP.Sid AND Pid='P2';
    
```

Diagram: A red arrow labeled "Join condition" points to the equals sign in the WHERE clause. Green arrows labeled "TableName.AttributeName" point from "S.Sid" and "SP.Sid" to their respective table and column names.

45

Join (n.1)

SP.Pid='P2'

S.Sid	S.SName	S.#Empl	S.City	SP.Sid	SP.Pid	SP.Qty
S1	Smith	20	London	S1	P1	300
S1	Smith	20	London	S1	P2	200
S1	Smith	20	London	S1	P3	400
S1	Smith	20	London	S1	P4	200
S1	Smith	20	London	S1	P5	100
S1	Smith	20	London	S1	P6	100
S2	Jones	10	Paris	S2	P1	300
S2	Jones	10	Paris	S2	P2	400
S3	Blake	30	Paris	S3	P2	200
S4	Clark	20	London	S4	P3	200
S4	Clark	20	London	S4	P4	300
S4	Clark	20	London	S4	P5	400

46

Join (n.1)

S.Sid	S.SName	S.#Empl	S.City	SP.Sid	SP.Pid	SP.Qty
S1	Smith	20	London	S1	P2	200
S2	Jones	10	Paris	S2	P2	400
S3	Blake	30	Paris	S3	P2	200

↓

R
Smith
Jones
Blake

47

Join (n.1)

- Find the names of the suppliers that provide product P2
 - in relational algebra

The diagram shows a join operation between two tables, S and SP. The join is represented by a diamond symbol with a double-headed arrow. The output is a projection of the join result, denoted as $\pi_{S.SName}$. The join condition is $\sigma_{PId='P2'}$.

48

Join (n.1)

- Find the names of the suppliers that provide product P2
 - in relational algebra

```

SELECT SName
FROM S, SP
WHERE S.SId=SP.SId
AND PId='P2';
    
```

↔

```

SELECT SName
FROM S,SP
WHERE PId='P2' AND
S.SId=SP.SId;
    
```

- The result and efficiency are independent
 - from the order of the predicates in the WHERE clause
 - from the order of the tables in the FROM clause

49

SQL Declarability

- In relational algebra (procedural language) we define the order in which the operators are applied
- In SQL (declarative language) the best order is chosen by the optimizer independently
 - from the order of the conditions in the WHERE clause
 - from the order of the tables in the FROM clause

50

Join (n.2)

- Find the name of suppliers who provide at least one red product

```

SELECT SName
FROM S, SP, P
WHERE S.SId=SP.SId AND P.PId=SP.PId
AND Color='Red';
    
```

- FROM Clause with N Tables
 - at least N-1 join conditions in the WHERE clause

51

Join (n.2)

- Find the pairs of supplier codes such that both suppliers are based in the same city

```

SELECT SX.SId, SY.SId
FROM S AS SX, S AS SY
WHERE SX.City=SY.City;
    
```

SId	SName	#Employees	City
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

SId	SName	#Employees	City
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

52

Join (n.2)

- Find the pairs of supplier codes such that both suppliers are based in the same city

```

SELECT SX.SId, SY.SId
FROM S AS SX, S AS SY
WHERE SX.City=SY.City;
    
```

SX.SId	SY.SId
S1	S1
S1	S4
S2	S2
S2	S3
S3	S2
S3	S3
S4	S1
S4	S4
S5	S5

- The result includes
 - pairs of identical values
 - permutations of the same pairs of values

53

Join (n.2)

- Find the pairs of supplier codes such that both suppliers are based in the same city

```
SELECT SX.Sid, SY.Sid
FROM S AS SX, S AS SY
WHERE SX.City=SY.City AND
SX.Sid <> SY.Sid;
```

R

SX.Sid	SY.Sid
S1	S1
S1	S4
S2	S2
S2	S3
S3	S2
S3	S3
S4	S1
S4	S4
S5	S5

- It removes pairs of identical values

DBG

54

54

Join (n.2)

- Find the pairs of supplier codes such that both suppliers are based in the same city

```
SELECT SX.Sid, SY.Sid
FROM S AS SX, S AS SY
WHERE SX.City=SY.City AND
SX.Sid < SY.Sid;
```

R

SX.Sid	SY.Sid
S1	S4
S2	S3
S3	S2
S4	S1
S4	S5
S5	S4

R

SX.Sid	SY.Sid
S1	S4
S2	S3

- It eliminates the permutations of the same pairs of values

DBG

55

55

Join: alternative syntax

- Different types of join may be specified
 - outer join
- It allows differentiating between
 - join conditions and
 - tuple selection conditions

```
SELECT [DISTINCT] Attributes
FROM Table JoinType JOIN Table ON
JoinCondition
[WHERE TupleConditions];
```

JoinType = < INNER | [FULL | LEFT | RIGHT] OUTER >

DBG

56

56

INNER join

- Find the names of the suppliers that supply at least one red product

```
SELECT SName
FROM P INNER JOIN SP ON P.PId=SP.PId
INNER JOIN S ON S.SId=SP.SId
WHERE P.Color='Red';
```

DBG

57

57

OUTER join

- Find the codes and the names of the suppliers together with the codes of the products they provide, also including the suppliers that are not supplying any product

```
SELECT S.Sid, SName, PId
FROM S LEFT OUTER JOIN SP ON
S.SId=SP.SId;
```

S.Sid	S.SName	SP.SId
S1	Smith	P1
S1	Smith	P2
S1	Smith	P3
S1	Smith	P4
S1	Smith	P5
S1	Smith	P6
S2	Jones	P1
S2	Jones	P2
S3	Blake	P2
S4	Clark	P3
S4	Clark	P4
S4	Clark	P5
S5	Adams	NULL

DBG

58

58

Aggregate Functions

Introduction to SQL

DBG

59

59

Aggregate function

- It operates on a set of values
- It produces a single (aggregate) value as a result
- It is specified in the SELECT clause
 - non-aggregate attributes may not be specified at the same time
 - multiple aggregate functions may be specified simultaneously
- Aggregate functions are only evaluated once all predicates in the WHERE clause have been applied

DBG

60

60

Aggregate functions

COUNT: count of elements in a given attribute

SUM: sum of values for a given attribute

AVG: average of values for a given attribute

MAX: maximum value of a given attribute

MIN: minimum value of a given attribute

DBG

61

61

COUNT

- Counts the number of elements in a set
 - rows in a table
 - (possibly distinct) values for one or more attributes

`COUNT (<*[DISTINCT | ALL] ListOfAttributes >))`

- If the function argument is preceded by **DISTINCT**, it counts the number of distinct values of the argument

62

62

The COUNT function (n.1)

- Find the number of suppliers

```
SELECT COUNT(*)
FROM S;
```

S				R	
Sid	SName	#Employees	City		
S1	Smith	20	London	→	5
S2	Jones	10	Paris		
S3	Blake	30	Paris		
S4	Clark	20	London		
S5	Adams	30	Athens		

DBG

63

63

The COUNT function (n.2)

- Find the number of suppliers that supply at least one product

SP		
Sid	Pid	Qty
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P3	200
S4	P4	300
S4	P5	400

```
SELECT COUNT(*)
FROM SP;
```

R	
	12

- It counts the number of supplied products, not the suppliers

DBG

64

64

The COUNT function (n.2)

- Find the number of suppliers that supply at least one product

SP		
Sid	Pid	Qty
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P3	200
S4	P4	300
S4	P5	400

```
SELECT COUNT(Sid)
FROM SP;
```

R	
	12

- It still counts the number of supplied products, not the suppliers

DBG

65

65

The COUNT function (n.2)

- Find the number of suppliers that supply at least one product

SP

Sid	Pid	Qty
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P3	200
S4	P4	300
S4	P5	400

```
SELECT COUNT(DISTINCT Sid)
FROM SP;
```

→

R
4

- It counts the number of distinct suppliers

DBG

66

Aggregate functions and WHERE

- Aggregate functions are only evaluated once all predicates in the WHERE clause have been applied

67

Aggregate functions and WHERE

- Find the number of suppliers providing product P2

SP

Sid	Pid	Qty
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P3	200
S4	P4	300
S4	P5	400

```
SELECT COUNT(*)
FROM SP
WHERE Pid='P2';
```

→

Sid	Pid	Qty
S1	P2	200
S2	P2	400
S3	P2	200

→

R
3

- Aggregate functions are only evaluated once all predicates in the WHERE clause have been applied

DBG

68

SUM, MAX, MIN, AVG

- SUM, MAX, MIN and AVG
 - they allow an attribute or an expression as argument
- SUM and AVG
 - they only allow numeric type or time interval attributes
- MAX and MIN
 - they require an expression that can be ordered
 - may also be applied to character strings and time instants

69

The SUM function

- Find the overall quantity of supplied pieces for product P2

SP

Sid	Pid	Qty
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P3	200
S4	P4	300
S4	P5	400

```
SELECT SUM(Qty)
FROM SP
WHERE Pid='P2';
```

→

Sid	Pid	Qty
S1	P2	200
S2	P2	400
S3	P2	200

→

R
800

DBG

70

The GROUP BY operator

Introduction to SQL

DBG

71

70

71

GROUP BY

- Grouping clause
GROUP BY ListOfGroupingAttributes
- The order of grouping attributes is irrelevant
- In the SELECT statement **only**
 - attributes specified in the GROUP BY clause
 - aggregate functions
 are allowed to appear
- Attributes that are unambiguously determined by other attributes already present in the GROUP BY clause may be added *without altering the result*

72

Grouping

- For each product, find the overall quantity of supplied pieces

Sid	Pid	Qty
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S1	P3	400
S4	P3	200
S1	P4	200
S4	P4	300
S1	P5	100
S4	P5	400
S4	P6	100

→

Sid	Pid	Qty
S1	P1	300
S2	P1	300
S1	P2	200
S2	P2	400
S3	P2	200
S1	P3	400
S4	P3	200
S1	P4	200
S4	P4	300
S1	P5	100
S4	P5	400
S1	P6	100

→

Pid	
P1	600
P2	800
P3	600
P4	500
P5	500
P6	100

73

Grouping

- For each product, find the overall quantity of supplied pieces

Sid	Pid	Qty
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S1	P3	400
S4	P3	200
S1	P4	200
S4	P4	300
S1	P5	100
S4	P5	400
S1	P6	100

→

Sid	Pid	Qty
S1	P1	300
S2	P1	300
S1	P2	200
S2	P2	400
S3	P2	200
S1	P3	400
S4	P3	200
S1	P4	200
S4	P4	300
S1	P5	100
S4	P5	400
S1	P6	100

→

Pid	
P1	600
P2	800
P3	600
P4	500
P5	500
P6	100

SELECT Pid, SUM(Qty)
FROM SP
GROUP BY Pid;

74

GROUP BY and WHERE

- For each product, find the overall quantity of pieces supplied by suppliers based in Paris

Sid	SName	#Employees	City
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

→

Sid	Pid	Qty
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P3	200
S4	P4	300
S4	P5	400

75

GROUP BY and WHERE

- For each product, find the overall quantity of pieces supplied by suppliers based in Paris

```

SELECT ...
FROM SP, S
WHERE SP.Sid=S.Sid AND City='Paris'
    
```

76

GROUP BY and WHERE

- For each product, find the overall quantity of pieces supplied by suppliers based in Paris

S.Sid	S.SName	S.#Employees	S.City	SP.Sid	SP.Pid	SP.Qty
S1	Smith	20	London	S1	P1	300
S1	Smith	20	London	S1	P2	200
S1	Smith	20	London	S1	P3	400
S1	Smith	20	London	S1	P4	200
S1	Smith	20	London	S1	P5	100
S1	Smith	20	London	S1	P6	100
S2	Jones	10	Paris	S2	P1	300
S2	Jones	10	Paris	S2	P2	400
S3	Blake	30	Paris	S3	P2	200
S4	Clark	20	London	S4	P3	200
S4	Clark	20	London	S4	P4	300
S4	Clark	20	London	S4	P5	400

77

GROUP BY and WHERE

- For each product, find the overall quantity of pieces supplied by suppliers based in Paris

```

SELECT PId, SUM(Qty)
FROM SP, P
WHERE S.PId=S.SId AND City='Paris'
GROUP BY PId;
    
```

- Products that are not supplied by any supplier are not included in the result

DBG 78

78

GROUP BY and WHERE

- For each product, find the overall quantity of pieces supplied by suppliers based in Paris

SPPId	SPQty
P1	300
P2	400
P2	200

→

SP.PId	Qty
P1	300
P2	600

R

DBG 79

79

GROUP BY and SELECT

- For each product, find the code, the *name* and the overall supplied quantity

```

SELECT P.PId, PName, SUM(Qty)
FROM P, SP
WHERE S.PId=S.PId
GROUP BY P.PId, PName
    
```

- attributes that are unambiguously determined by other attributes already present in the GROUP BY clause may be added *without altering the result*

DBG 80

80

Group selection condition: HAVING

- You cannot use the WHERE clause to define selection conditions on groups
- Selection condition on groups expressed in HAVING clause:
 - HAVING** Group Conditions
 - it is possible to specify conditions *only* on aggregated functions

81

81

Group selection condition(n.1)

- Find the overall quantity of supplied pieces for the products for which at least 600 pieces are supplied *overall*

SId	PId	Qty
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P3	200
S4	P4	300
S4	P5	400
S4	P6	100

→

SId	PId	Qty
S1	P1	300
S2	P1	300
S1	P2	200
S2	P2	400
S3	P2	200
S1	P3	400
S4	P3	200
S1	P4	200
S4	P4	300
S1	P5	100
S4	P5	400
S1	P6	100

→

PId	Qty
P1	600
P2	800
P3	600

R

DBG 82

82

Group selection condition (n.1)

- Find the overall quantity of supplied pieces for the products for which at least 600 pieces are supplied *overall*

```

SELECT PId, SUM(Qty)
FROM SP
GROUP BY PId
HAVING SUM(Qty)>=600;
    
```

- The **HAVING** clause allows the specification of conditions on the aggregate functions

DBG 83

83

Group selection condition (n.2)

- Find the codes of the red products supplied by more than one supplier

P

PId	PName	Color	Size	Store
P1	Jumper	Red	40	London
P2	Jeans	Green	48	Paris
P3	Blouse	Blue	48	Rome
P4	Blouse	Blue	44	London
P5	Skirt	Blue	40	Paris
P6	Shorts	Red	42	London

SP

SId	PId	Qty
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P3	200
S4	P4	300
S4	P5	400

DBG

84

84

Group selection condition(n.2)

- Find the codes of the red products supplied by more than one supplier

```
SELECT SP.PId
FROM SP, P
WHERE SP.PId=P.PId AND Color='Red'
GROUP BY SP.PId
HAVING COUNT(*)>1;
```

DBG

85

85

Group selection condition (n.2)

- Find the codes of the red products supplied by more than one supplier

S.SId	S.PId	S.Qty	P.PId	P.PName	P.Color	P.Size	P.Store
S1	P1	300	P1	Jumper	Red	40	London
S2	P1	300	P1	Jumper	Red	40	London
S1	P6	100	P6	Shorts	Red	42	London



R

PId
P1

DBG

86

86