Data Management and Warehousing

Sorting and Counting

Stéphane Bressan

NUS
National University
of Singapore





We want to develop a sales analysis application for our online gaming store. We would like to store several items of information about our customers: their first name, last name, date of birth, e-mail, date and country of registration on our online sales service and the customer identifier that they have chosen . We also want to manage the list of our products, the games, their version and price. The price is fixed for each version of each game. Finally, our customers buy and download games. So we must remember which version of which game each customer has downloaded. It is not important to keep the download date for this application.

The order of the records in the tables is in never guaranteed. However, you can sort the results of a query by adding the "ORDER BY" clause followed by a list of fields. The following query displays the last name and first name of registered clients from Singapore in descending alphabetical order of the first names. The keyword to get an ascending order is "ASC". This is the default mode and the keyword is often omitted. For descending order, add the keyword "DESC".

```
SELECT c.last_name, c.first_name
FROM customers c
WHERE c.country = 'Singapore'
ORDER BY c.first_name ASC;
```

last_name	first_name
Griffin	Aaron
Green	Adam
Stone	Adam
Romero	Adam
Wijaya	Adam
Hansen	Alan
Richards	Alan
Porter	Alan
Khoo	Albert
Arnold	Albert
Torres	Albert
Dean	Albert
. . .	

It is possible to indicate a list of fields (or of calculated fields) in the "ORDER BY" clause. Additional fields will be used to sort records that have the same value for previous fields. The order of fields in the list is important. The query below displays the name, version and price of the games in ascending alphabetical order of the names and alphanumeric order of the versions. Two games with the same names are ordered according to their version.

```
SELECT g.name, g.version, g.price
FROM games g
ORDER BY g.name ASC, g.version DESC;
```

name	version	price
Aerified	3.0	3.99
Aerified	2.1	12
Aerified	2.0	5
Aerified	1.2	1.99
Aerified	1.1	3.99
Aerified	1.0	12
Alpha	3.0	5
Alpha	2.1	2.99
Alpha	2.0	12
. . .		

It is possible to sort the displayed result of a query according to fields that are not displayed. The query below displays the name of the games in descending numerical order of their price (from the most expensive to the cheapest). Note that the order fits the domain.

```
SELECT g.name FROM games g
ORDER BY g.price DESC;
```

This is why you can not use "`DISTINCT`" with "`ORDER BY`".

name
Duobam
It
Stim
Cookley
Daltfresh
Daltfresh
Daltfresh
Domainer
Sonsing
Domainer
Sonair
Redhold
Duobam
…

NUS National University of Singapore

Exercise

Print the email and the name of customers and the version of the games that they downloaded in alphabetical ascending order of customer names and in descending numerical order of game prices.

The values of a column can be aggregated in groups using the "GROUP BY" clause followed by a list of fields and aggregation functions such as "COUNT()", "SUM()", "MAX()", "MIN()", " AVG()", "STD()"and so on. "GROUP BY" creates groups by grouping records that have the same value for the fields indicated. The aggregation functions are calculated for each group. The following query displays the countries from which customers have registered and the number of customers for each country.

```
SELECT c.country, COUNT(c.customerid)
FROM customers c
GROUP BY c.country;
```

Since we are counting rows we need not indicate the field.

```
SELECT c.country, COUNT(*)
FROM customers c
GROUP BY c.country;
```

country	first_name	last_name	email	dob	since	customerid
Indonesia	Cheryl	Reyes	creyesjl@jalbum.net	1/9/1992	2/22/2016	Cheryl1992
Indonesia	Marie	Flores	mfloresk2@sogou.com	10/4/1997	11/25/2016	Marie1997
Indonesia	Marie	Young	myoung33@goo.ne.jp	9/13/1996	10/24/2016	Marie1996
…						
Malaysia	Johnny	Gilbert	jgilberte8@nymag.com	6/19/1989	1/15/2016	JohnnyG89
Malaysia	Cynthia	Pierce	cpierce25@prlog.org	10/14/1990	11/20/2016	Cynthia1990
Malaysia	Nicole	Lee	nleef1@whitehouse.gov	9/23/1984	3/21/2016	NicoleL84
…						
Singapore	Joseph	Mao	jmaor6@bizjournals.com	8/18/1984	3/15/2016	JosephM84
Singapore	Karen	Graham	kgraham76@phpbb.com	5/28/1981	12/26/2016	KarenG81
Singapore	Katherine	Kuningan	kkuningan7f@redcross.org	3/1/1982	1/9/2016	KatherineC82
…						
Thailand	Jean	Ling	jlingpn@walmart.com	4/4/1990	11/25/2016	Jean90
Thailand	Raymond	Tan	rtan1z@nature.com	9/16/1993	6/23/2016	Raymond1993
Thailand	Jean	Nichols	jnichols52@dmoz.org	6/24/1984	4/21/2016	JeanN84
…						
France	Carole	Yoga	cyoga@glarge.org	8/1/1989	9/15/2016	Carole89
Vietnam	Gerald	Ford	gfordij@zdnet.com	10/7/1982	4/16/2016	GeraldF82
Vietnam	Fred	Fields	ffieldsdl@ask.com	1/16/1985	11/29/2016	FredF85
Vietnam	Russell	Hakim	rhakim7y@si.edu	6/24/2000	9/28/2016	Russell2000
…						

Indonesia

Malaysia

Singapore

Thailand

France

Vietnam

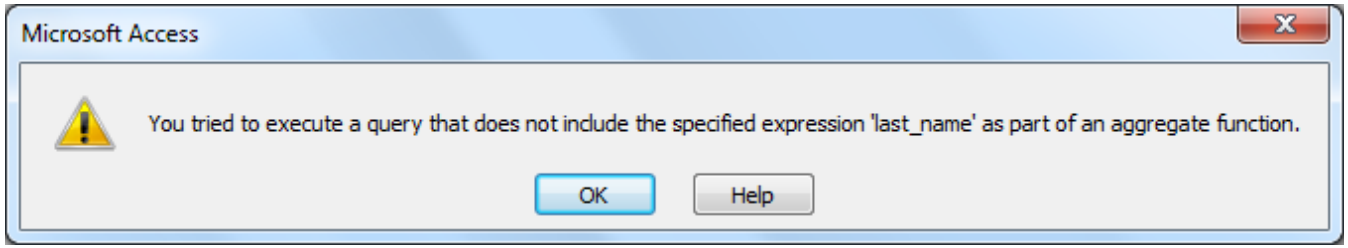
```
GROUP BY
c.country
```

The RDBMS creates the groups by grouping records that have the same values for the specified fields before computing the aggregate functions.

country	Expr1001
France	1
Indonesia	243
Malaysia	168
Singapore	391
Thailand	100
Vietnam	98

It is only possible to display the fields present in the "GROUP BY" clause and the aggregation functions because other fields do not necessarily have the same value.

```
SELECT c.country, c.last_name, COUNT(*)
FROM customers c
GROUP BY c.country;
```



It is possible to specify a list of fields (or calculated fields) in the "GROUP BY" clause. "GROUP BY" creates groups by grouping records that have the same value for the fields indicated. the order of fields in the list does not matter. The query below shows the number of downloads by country of registration and year of birth of customers.

```
SELECT c.country, EXTRACT(YEAR FROM c.since) AS
year, COUNT(*) AS total
FROM customers c INNER JOIN downloads d ON
c.customerid = d.customerid
GROUP BY c.country, EXTRACT(YEAR FROM c.since);
```

country	year	total
Indonesia	2015	5
Indonesia	2016	998
Malaysia	2015	3
Malaysia	2016	713
Singapore	2015	4
Singapore	2016	1669
Thailand	2015	6
Thailand	2016	414
Vietnam	2015	3
Vietnam	2016	399

"COUNT(*)" counts the records in the group. "COUNT (<field>)" counts the field values in the group (records that have the same value for the specified field). In RDBMS other than Microsoft Access 2010, you can use "COUNT (DISTINCT <field>)" if you do not want to count duplicate values. The queries below show the number of client names (the number of clients) and the number of different client names, respectively. Note that there is no "GROUP BY" clause. The default group is the entire table.

```
SELECT COUNT(last_name)
FROM customers c;

SELECT COUNT(DISTINCT last_name)
FROM customers c;
```

There are other aggregate functions than "COUNT ( )". The domain of the field must be compatible with the operation (that is, generally, of numeric or date type but some aggregate functions work for other domains). The query below displays the first and last name of the users and the total price of the downloaded games for each user. Note that users who have not downloaded games do not appear in the result. Note also that you must add fields in the "GROUP BY" clause to be able to display them.

```
SELECT c.first_name, c.last_name, sum(g.price) AS
spending
FROM (customers AS c
INNER JOIN downloads AS d
ON c.customerid = d.customerid)
INNER JOIN games AS g
ON d.name = g.name AND d.version = g.version
GROUP BY c.customerid, c.first_name, c.last_name;
```

first_name	last_name	spending
Adam	Green	8.97
Adam	Stone	22
Adam	Howell	28.98
Adam	Wijaya	37.96
Alan	Richards	20.99
Alan	Cruz	29.98
Alan	Johnson	22
Alan	Walker	5.98
Alan	Porter	5
Alan	Mendoza	29
…		

Exercise

Print the name and version of the games in ascending numerical order of the number of their download. Games that have never been downloaded will be ignored.

Aggregate functions can be used in the "ORDER BY" clause.

Aggregate functions can be used to filter the displayed records. This cannot be done in the "WHERE" clause since the conditions in this clause are applied before the groups are created. We use the "HAVING" clause that filters the records after the groups are created. There is therefore no aggregate function in the "WHERE" clause. There are only fields used for creating groups and aggregate functions in the "HAVING" clause. The following query displays the countries from which 100 or more customers have registered.

```
SELECT c.country
FROM customers c
GROUP BY c.country
HAVING COUNT(*) >= 100;
```

country
Indonesia
Malaysia
Singapore
Thailand

Exercise

Which games are most downloaded?

Aggregation functions can be used in subqueries and subqueries in the "HAVING" clause.

Exercise

What are our best customers? (those who spent the most money).

NUS National University of Singapore

Credits

The content of this lecture is based on chapter 3 of the book "Introduction to database Systems"
By
S. Bressan and B. Catania, McGraw Hill publisher

Images and clips used in this presentation are licensed from Microsoft Office Online Clipart and Media

For questions about the content of this course and about copyrights, please contact Stéphane Bressan

steph@nus.edu.sg