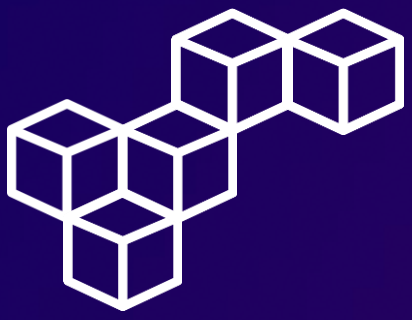
AWS re:Start LAB

Organizing Data

WEEK 7

aws re/start



Overview

Organizing data is crucial for extracting meaningful insights and making informed decisions. The GROUP BY clause with aggregate functions like SUM( ) enables the categorization of data based on specific criteria, facilitating the calculation of summary statistics and trends. This approach is essential for summarizing data at various levels and gaining a comprehensive understanding of dataset characteristics.

Moreover, utilizing the OVER clause with window functions such as RANK( ) enhances data organization by ranking data within defined windows. This functionality allows for the identification of top performers or outliers, supporting performance evaluation and anomaly detection. Combining these organizational techniques with aggregate functions and window functions provides a powerful framework for organizing and analyzing data, ultimately empowering businesses to derive actionable insights and drive data-centric decision-making processes.

Topics covered

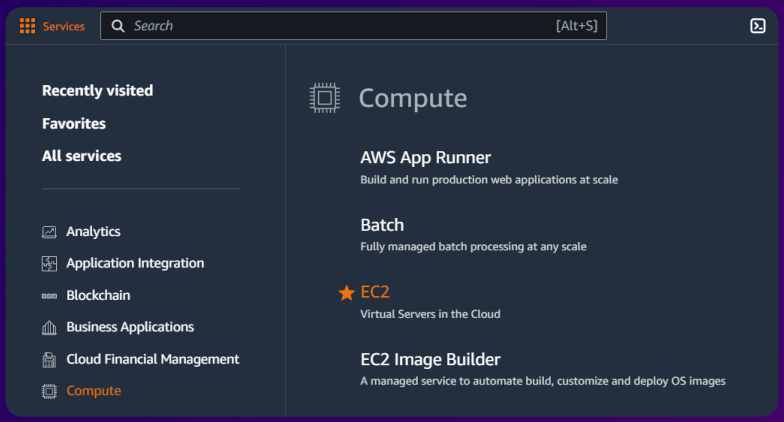
- Use the GROUP BY clause with the aggregate function SUM( )
- Use the OVER clause with the RANK( ) window function
- Use the OVER clause with the aggregate function SUM( ) and the RANK( ) window function

Task 1

Connect to the Command Host

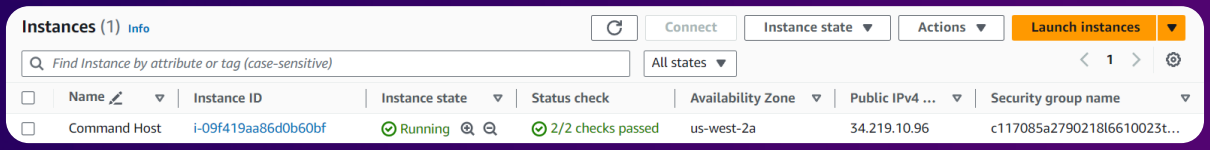
Step 1: Access the EC2 Management Console

Open the AWS Management Console, and select EC2.



Step 2: Review running instances

Navigate to the **Instances** section. The running **Command Host** instance is listed.



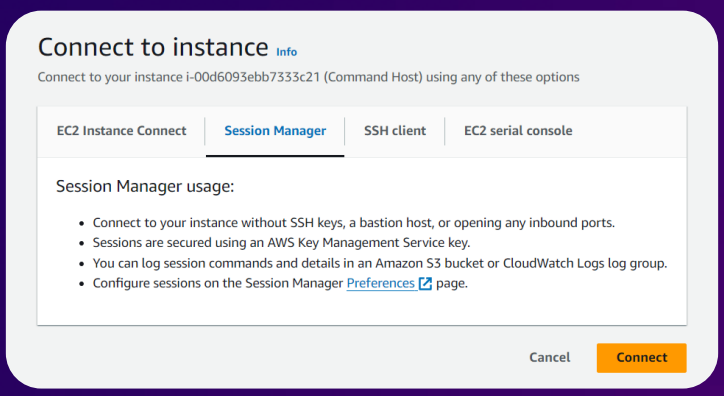
aws re/start

Task 1

Connect to the Command Host

Step 3: Connect to the instance

Connect to the **Command Host** EC2 instance, which contains a database client, using Session Manager.



Step 4: Connect to the database server

To connect to the database server, run the following commands in the terminal.

```
sh-4.2$ sudo su
[root@ip-10-1-11-180 bin]# cd /home/ec2-user/
[root@ip-10-1-11-180 ec2-user]# mysql -u root --password='re:St@rt!9'
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 14
Server version: 10.6.17-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

aws re/start

Task 2

Query the world database

Step 1: Show existing databases

Show the existing databases using the SHOW DATABASES query, and verify that a database named **world** is available.

```
MariaDB [(none)]> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| world              |
+--------------------+
5 rows in set (0.003 sec)
```

Step 2: Review the table schema

Review the table schema, data, and number of rows in the **country** table, using the SELECT * FROM statement.

```
MariaDB [(none)]> SELECT * FROM world.country;
```

Code	Name	Continent	Region	SurfaceArea	IndepYear	Population	LifeExpectancy	GNP	GNPOld	LocalName	GovernmentForm	Capital	Code2
ABW	Aruba	North America	Caribbean	193.00	NULL	103000	78.4	828.00	793.00	Aruba	Nonmetropolitan Territory of The Netherlands	129	AW
AFG	Afghanistan	Asia	Southern and Central Asia	652090.00	1919	22720000	45.9	5976.00	NULL	Afganistan/Afqanestan	Islamic Emirate	1	AF
AGO	Angola	Africa	Central Africa	1246700.00	1975	12878000	38.3	6648.00	7984.00	Angola	Republic	56	AO
AIA	Anguilla	North America	Caribbean	96.00	NULL	8000	76.1	63.20	NULL	Anguilla	Dependent Territory of the UK	62	AI
ALB	Albania	Europe	Southern Europe	28748.00	1912	3401200	71.6	3205.00	2500.00	Shqipëria	Republic	34	AL
AND	Andorra	Europe	Southern Europe	468.00	1278	78000	83.5	1630.00	NULL	Andorra	Parliamentary Coprincipality	55	AD
ANT	Netherlands Antilles	North America	Caribbean	800.00	NULL	217000	74.7	1941.00	NULL	Nederlandse Antillen	Nonmetropolitan Territory of The Netherlands	33	AN
ARE	United Arab Emirates	Asia	Middle East	83600.00	1971	2441000	74.1	37966.00	36846.00	Al-Imarat al-´Arabiya al-Muttahida	Emirate Federation	65	AE
ARG	Argentina	South America	South America	2780400.00	1816	37032000	75.1	340238.00	323310.00	Argentina	Federal Republic	69	AR
ARM	Armenia	Asia	Middle East	29800.00	1991	3520000	66.4	1813.00	1627.00	Hajastan	Republic	126	AM
ASM	American Samoa	Oceania	Polynesia	199.00	NULL	68000	75.1	334.00	NULL	Amerika Samoa	US Territory	54	AS
ATA	Antarctica	Antarctica	Antarctica	13120000.00	NULL	0	NULL	0.00	NULL	–	Co-administrated	NULL	AQ
ATF	French Southern territories	Antarctica	Antarctica	7780.00	NULL	0	NULL	0.00	NULL	Terres australes françaises	Nonmetropolitan Territory of France	NULL	TF
ATG	Antigua and Barbuda	North America	Caribbean	442.00	1981	68000	70.5	612.00	584.00	Antigua and Barbuda	Constitutional Monarchy	63	AG
AUS	Australia	Oceania	Australia and New Zealand	7741220.00	1901	18886000	79.8	351182.00	392911.00	Australia	Constitutional Monarchy, Federation	135	AU
AUT	Austria	Europe	Western Europe	83859.00	1918	8091800	77.7	211860.00	206025.00	Österreich	Federal Republic	1523	AT
AZE	Azerbaijan	Asia	Middle East	86600.00	1991	7734000	62.9	4127.00	4100.00	Azärbaycan	Federal Republic	144	AZ
BDI	Burundi	Africa	Eastern Africa	27834.00	1962	6695000	46.2	903.00	982.00	Burundi/Uburundi	Republic	552	BI
BEL	Belgium	Europe	Western Europe	30518.00	1830	10239000	77.8	249704.00	243948.00	België/Belgique	Constitutional Monarchy, Federation	179	BE
BEN	Benin	Africa	Western Africa	112622.00	1960	6097000	50.2	2357.00	2141.00	Bénin	Republic	187	BJ
BFA	Burkina Faso	Africa	Western Africa	274000.00	1960	11937000	46.7	2425.00	2201.00	Burkina Faso	Republic	549	BF
BGD	Bangladesh	Asia	Southern and Central Asia	143998.00	1971	129155000	60.2	32852.00	31966.00	Bangladesh	Republic	150	BD
BGR	Bulgaria	Europe	Eastern Europe	110994.00	1908	8190900	70.9	12178.00	10169.00	Balgarija	Republic	539	BG
BHR	Bahrain	Asia	Middle East	694.00	1971	617000	73.0	6366.00	6097.00	Al-Bahrayn	Monarchy (Emirate)	149	BH
BHS	Bahamas	North America	Caribbean	13878.00	1973	307000	71.1	3527.00	3347.00	The Bahamas	Constitutional Monarchy	148	BS
BIH	Bosnia and Herzegovina	Europe	Southern Europe	51197.00	1992	3972000	71.5	2841.00	NULL	Bosna i Hercegovina	Federal Republic	201	BA
BLR	Belarus	Europe	Eastern Europe	207600.00	1991	10236000	68.0	13714.00	NULL	Belarus	Republic	3520	BY
BLZ	Belize	North America	Central America	22696.00	1981	241000	70.9	630.00	616.00	Belize	Constitutional Monarchy	185	BZ
BMU	Bermuda	North America	North America	53.00	NULL	65000	76.9	2328.00	2190.00	Bermuda	Dependent Territory of the UK	191	BM
BOL	Bolivia	South America	South America	1098581.00	1825	8329000	63.7	8571.00	7967.00	Bolivia	Republic	194	BO
BRA	Brazil	South America	South America	8547403.00	1822	170115000	62.9	776739.00	804108.00	Brasil	Federal Republic	211	BR
BRB	Barbados	North America	Caribbean	430.00	1966	270000	73.0	2223.00	2186.00	Barbados	Constitutional Monarchy	174	BB
BRN	Brunei	Asia	Southeast Asia	5765.00	1984	328000	73.6	11705.00	12460.00	Brunei Darussalam	Monarchy (Sultanate)	538	BN
BTN	Bhutan	Asia	Southern and Central Asia	47000.00	1910	2124000	52.4	372.00	383.00	Druk-Yul	Monarchy	192	BT
BVT	Bouvet Island	Antarctica	Antarctica	59.00	NULL	0	NULL	0.00	NULL	Bouvetøya	Dependent Territory of Norway	NULL	BV
BWA	Botswana	Africa	Southern Africa	581730.00	1966	1622000	39.3	4834.00	4935.00	Botswana	Republic	204	BW
CAF	Central African Republic	Africa	Central Africa	622984.00	1960	3615000	44.0	1054.00	993.00	Centrafrique/Bê-Afrîka	Republic	1889	CF
CAN	Canada	North America	North America	9970610.00	1867	31147000	79.4	598862.00	625626.00	Canada	Constitutional Monarchy, Federation	1822	CA

aws re/start

Task 2

Query the world database

Step 3: Use the ORDER BY clause

Return a list of records where the **Region** is 'Australia and New Zealand'. Use the ORDER BY clause to arrange the results by **Population** in descending order.

```
MariaDB [(none)]> SELECT Region, Name, Population FROM world.country WHERE Region = 'Australia and New Zealand'
 ORDER BY Population DESC;
+--------------------------+-------------------------+------------+
| Region                   | Name                    | Population |
+--------------------------+-------------------------+------------+
| Australia and New Zealand | Australia               |   18886000 |
| Australia and New Zealand | New Zealand             |    3862000 |
| Australia and New Zealand | Christmas Island        |       2500 |
| Australia and New Zealand | Norfolk Island          |       2000 |
| Australia and New Zealand | Cocos (Keeling) Islands |        600 |
+--------------------------+-------------------------+------------+
5 rows in set (0.001 sec)
```

Step 4: Use the GROUP BY clause

Filter records using a condition where the **Region** is equal to 'Australia and New Zealand'. Group the results using a GROUP BY clause and return a SUM( ) of the **Population**.

```
MariaDB [(none)]> SELECT Region, SUM(Population) FROM world.country WHERE Region = 'Australia and New Zealand'
 GROUP BY Region ORDER BY SUM(Population) DESC;
+--------------------------+-----------------+
| Region                   | SUM(Population) |
+--------------------------+-----------------+
| Australia and New Zealand |        22753100 |
+--------------------------+-----------------+
1 row in set (0.001 sec)
```

Task 2

Query the world database

Step 5: Use the OVER( ) clause

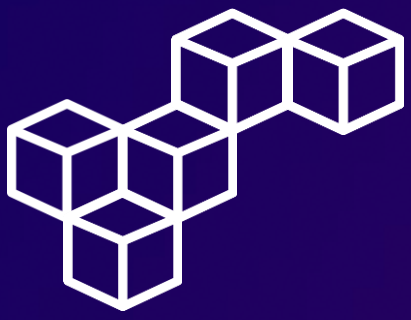
Use the OVER( ) clause to group the records by **Region** and use the SUM( ) function to aggregate the records and generate a running total by adding the **Population**.

```
MariaDB [(none)]> SELECT Region, Name, Population, SUM(Population) OVER(PARTITION BY Region ORDER BY Population)
 AS 'Running Total' FROM world.country WHERE Region = 'Australia and New Zealand';
+--------------------------+-------------------------+------------+---------------+
| Region                   | Name                    | Population | Running Total |
+--------------------------+-------------------------+------------+---------------+
| Australia and New Zealand | Cocos (Keeling) Islands |        600 |           600 |
| Australia and New Zealand | Norfolk Island          |       2000 |          2600 |
| Australia and New Zealand | Christmas Island        |       2500 |          5100 |
| Australia and New Zealand | New Zealand             |    3862000 |       3867100 |
| Australia and New Zealand | Australia               |   18886000 |      22753100 |
+--------------------------+-------------------------+------------+---------------+
5 rows in set (0.001 sec)
```

Step 6: Use the RANK( ) function

Group the records by **Region** and order them by **Population** with the OVER( ) clause. Use the RANK( ) function to generate a rank number indicating the position of each record in the result set.

```
MariaDB [(none)]> SELECT Region, Name, Population, SUM(Population) OVER(PARTITION BY Region ORDER BY Population) AS 'Running Total',
 RANK() OVER(PARTITION BY Region ORDER BY Population) AS 'Ranked' FROM world.country WHERE Region = 'Australia and New Zealand';
+--------------------------+-------------------------+------------+---------------+--------+
| Region                   | Name                    | Population | Running Total | Ranked |
+--------------------------+-------------------------+------------+---------------+--------+
| Australia and New Zealand | Cocos (Keeling) Islands |        600 |           600 |      1 |
| Australia and New Zealand | Norfolk Island          |       2000 |          2600 |      2 |
| Australia and New Zealand | Christmas Island        |       2500 |          5100 |      3 |
| Australia and New Zealand | New Zealand             |    3862000 |       3867100 |      4 |
| Australia and New Zealand | Australia               |   18886000 |      22753100 |      5 |
+--------------------------+-------------------------+------------+---------------+--------+
5 rows in set (0.001 sec)
```



Conclusions

The ORDER BY clause

The ORDER BY clause is essential for sorting query results in ascending or descending order based on specified columns, improving data presentation and analysis.

The GROUP BY clause

The GROUP BY clause is crucial for categorizing data into groups based on common attributes, enabling the calculation of aggregate functions like SUM( ) for each group.

The SUM( ) function

The SUM( ) function facilitates the calculation of total values within a dataset, providing valuable insights into numerical data and supporting aggregation operations.

The OVER( ) clause

The OVER( ) clause, when used with window functions like RANK( ), empowers users to perform advanced data analysis by ranking data within defined partitions or windows.

The RANK( ) window function

The RANK( ) window function allows for the ranking of data based on specified criteria within a dataset, aiding in performance evaluation and identifying top performers or outliers.

aws re/start

Cristhian Becerra

cristhian-becerra-espinoza

+51 951 634 354

cristhianbecerra99@gmail.com

Lima, Peru