Migrating a Database to Amazon RDS

Step 1: Create an RDS instance

1. Create an RDS instance that complies with these specifications.
    - **Templates**: *Dev/Test*
    - **DB instance identifier**: CafeDatabase
    - **Username**: admin
    - **Password**: Caf3DbPassw0rd!
    - **DB Instance Class**: *db.t3.micro*
    - **Storage type**: *General Purpose (SSD)*
    - **Allocated storage**: 20 GiB
    - Do *not* create a standby instance
    - Place it in the **Lab VPC**
    - **Subnet Group**: lab-db-subnet-group, where the database is *not* publicly accessible.
    - Choose existing **VPC security group** named dbSG, and *unselect* the default security group.
    - **Availability Zone**: Choose the first Availability Zone in the list, which ends in a. For example, if the Region is *us-east-1*, choose **us-east-1a**.
    - **Database port**: Keep the default TCP port of *3306*.

Step 2: Analyzing the existing café application deployment

Connect to the existing EC2 instance that runs the current café application.

6. Browse to the EC2 Console and choose **Running instances**. Notice the running instance named **CafeServer**.
7. Test the café application.
    - Open a new browser tab and load the café application at http://<public-ip-address>/cafe.
    - Browse to the **Menu** page and test placing an order.
      To do this, change the quantity for at least one menu item to at least *1* and choose **Submit Order**. An **Order Confirmation** page should display, which indicates that the café website is working as intended.
    - Choose **Order History**.
      The page shows that many orders were placed. The current database contains past customer orders that you will migrate to a database that's hosted on Amazon RDS.
8. Connect to the EC2 instance by using AWS Systems Manager to access a terminal session in the browser.
    - Navigate to the **Systems Manager** Console and choose **Session Manager**.
    - Start a session and connect to the **CafeServer**. You should now have a new browser tab open, with a terminal session that's connected to the EC2 instance.
    - At the prompt, enter the following commands:

```
bash
sudo su
su ec2-user
whoami
cd /home/ec2-user/
```

- The first command gave you a Bash shell. The second command switched your session to use the root user account on the EC2 instance. The third command switched you to use the *ec2-user* account. The fourth command should have returned output that confirms that you are connected as the ec2-user. The last command switches your terminal to the home directory of the ec2-user.

Business Requirement: Export data from the old database and establish a connection to the new database.

Step 3: Working with the database on the EC2 instance

9. Observe details of the database that runs on the EC2 instance.In the terminal, run these commands:

```
service mariadb status
mysql --version
```

10. The output should show that the locally installed MariaDB database on this EC2 instance is running. It should also show the version number of the database. Leave this browser tab open.
10. Return to the browser tab with the AWS Systems Manager console open in it.
11. From the panel on the left, under Application Management, choose Parameter Store. There are seven parameters stored. The café application PHP code references these values—for example, to retrieve the connection information for the database.
    - Choose the /cafe/dbPassword parameter, and copy the Value to your clipboard. You will use this value in a moment.
12. Connect to the database that is running on the EC2 instance. In the browser tab with the Bash terminal, connect the terminal-based MySQL client to the database by running this command:

```
mysql -u root -p
```

13. When prompted for the database password, paste the *dbPassword* parameter value that you copied a moment ago. You should now see a mariadb> prompt. This prompt indicates that you are now connected to the MariaDB database that runs on this EC2 instance.

14. Observe the data in the existing database. To observe the contents of the database, enter the following commands. In particular, you will review the tables that support the café web application.

```
show databases;
use cafe_db;
show tables;
select * from `order`;
```

15. These commands show all the orders that were placed, including the order that you placed a moment ago.

```
select * from `order_item`;
```

16. This command shows the order line items. Each order number has a row for each type of item that was ordered, with details about the quantity of each item and the price. All this data must be migrated to the new database.
17. Exit the SQL client.

```
exit;
```

15. Capture existing data in a file by using the *mysqldump* utility.
16. mysqldump --databases cafe_db -u root -p > CafeDbDump.sql
17. When prompted for the database password, paste the *dbPassword* value from the Systems Manager Parameter Store.
16. Confirm that *mysqldump* succeeded.
    - Run the ls command in the terminal. The output should show that the CafeDbDump.sql file was created.
    - Run the cat CafeDbDump.sql command to see the contents of the file.

Step 4: Working with the RDS database

In this task, you will first answer a few questions about the RDS instance that you created. Then, you will confirm that you can connect to the RDS instance.

17. In the AWS Management Console, return to the RDS service console and confirm that the *cafedatabase* RDS instance you created is now available.
18. Establish a network connection from the terminal running on the EC2 instance to the new RDS instance.
19. Run the show databases; command.
20. Notice that *cafe_db* database is not in the list yet. This situation is expected, because you haven't imported any data.
21. To disconnect, run the exit; command.

Business Requirements: Import data and connect to the new database.

Step 5: Importing the data into the RDS database instance

22. Import the data that you exported in task 3 to the RDS database instance.
    - To import the data, in the terminal, run the following command (where <rds-endpoint> is the actual endpoint):

```
mysql -u admin -p --host <rds-endpoint> < CafeDbDump.sql
```

    - At the password prompt, enter the password for the RDS instance. If you don't see any errors, the command likely succeeded.
23. Confirm that the data was imported.
    - To connect to the RDS database, run this command:

```
mysql -u admin -p --host <rds-endpoint>
```

    - At the password prompt, enter the password for the RDS instance.
    - To confirm that the data was imported, run the following command:

```
show databases;
use cafe_db;
show tables;
select * from `order`;
```

The output of the *select* statement should show at least 24 orders in the database.

    - Exit the SQL client:

```
Exit;
```

Step 6: Connecting the café application to the new database

24. Return to the AWS Systems Manager console browser tab.
25. From the panel on the left, choose Parameter Store. Recall that the café application's PHP code references these values.
26. Connect the café application to the RDS instance.Because the database connection information has changed, you must update these values to connect the application to the new RDS database instance instead of to the database running on the EC2 instance.
26. Confirm that your web application now uses the new database.

- Stop the database that's still running on the EC2 instance. In the terminal, use this command:

```
sudo service mariadb stop
```

- Load the http://<public-ip>/cafe/menu.php page and confirm that the application still works by placing an order.
- Choose Order History. Your latest order—and all the other previous orders—should be there. These orders are the data that you migrated to the new database.