

Proyecto Final – Ciencia de Datos en Python

Proyecto: Ingeniería de Datos con Python

Tema: Python, Pandas, SQL, ETL, AWS

Fecha y Hora de Entrega: 12/04/2024 23:55

Formato de Entrega: Archivos de Construcción y Video.

Grupo: Grupos de 2 o 3 personas

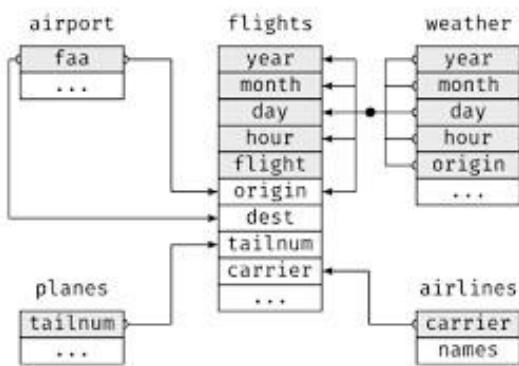
Calificación: Presentación por medio de Vídeo

ESTUDIANTE: Pedro Cabrera Juárez

Carnet: 20037437

DESCRIPCIÓN: Para este proyecto usted deberá desarrollar un pipeline de ingeniería de datos utilizando Python, SQL y AWS como herramientas de desarrollo, su proyecto tener los siguientes componentes:

Sistema transaccional: El cual será construido utilizando una instancia de RDS en AWS y SQL.



Para este sistema transaccional deberá crear el código SQL necesario considerando los tipos de datos adecuados, llaves primarias y llaves foráneas.

Ingestión de datos transaccionales: Deberá poblar la base de datos utilizando a partir de los archivos de .CSV para esto deberá usar Python y librerías de conexión de Python a SQL.

Preguntas de Negocio: Deberá plantear 10 preguntas de negocio que se puedan resolver utilizando la estructura de datos propuesta por el proyecto.

ETL y Analytics: Utilizando Markdown, numpy, pandas, matplotlib y seaborn, deberá elaborar un notebook con las respuestas a la preguntas planteadas de forma consistente y lo más profundo que sea posible. Para este caso deberá mostrar el código necesario para resolver cada pregunta planeada.

Adicionalmente deberá responder las siguientes preguntas de negocio utilizando tablas y gráficas:

1. ¿En qué país y qué avión se encuentra entre el 85% y el 70% de la cantidad de aterrizajes? Proporcione el nombre del país y el nombre del avión.
2. ¿Indique cuál es el aeropuerto con la temperatura más alta registrada en los datos?
3. ¿Cuál es la área de línea con la menor cantidad de vuelos registrados, indique cuantos vuelos, el código de la área de línea, el nombre completo?
4. Indique la media, mediana, mínimo, máximo y desviación estándar de las millas recorridas por cada avión, debe mostrar el nombre del avión y la información estadística en columnas adicionales
5. Muestre un cubo de información incluyendo la información de todas las tablas proporcionadas.
 - a. Indique la cantidad de filas y columnas
 - b. Indique cuantas y cuales son las variables categóricas, continuas, discretas y de fecha y hora.
 - c. Muestre una gráfica
 - i. de barras para la cantidad de las variables categóricas y discretas.
 - ii. De densidad para las variables continuas.
 - iii. Serie de tiempo con el conteo de apariciones para las de fecha y hora.

DETALLES TECNICOS: A continuación se describen los detalles técnicos mínimos que su proyecto debe cumplir:

- Para desarrollar el sistema transaccional podrá utilizar cualquier gestor de base de datos SQL que esté disponible en RDS.

aws

Iniciar sesión como usuario de IAM

ID de cuenta (12 dígitos) o alias de cuenta
533267100860

Nombre de usuario:
papd_A

Contraseña:
.....

Recordar esta cuenta

Iniciar sesión

[Iniciar sesión con el email del usuario raíz](#)

[Olvidó la contraseña?](#)

Amazon Lightsail

Lightsail es la manera más fácil de empezar a usar AWS

[Más información »](#)



aws

Autenticación multifactor

Escriba un código de MFA para completar el inicio de sesión.

Código de MFA:
740153

Enviar

[Cancelar](#)

RDS | us-east-2

us-east-2.console.aws.amazon.com/rds/home?region=us-east-2#launch-dbinstance:

YouTube Maps Gmail Azure - Sign up Course overview | G..

Todos los marcadores

AWS Servicios Buscar [Alt+S]

Elegir un método de creación de base de datos [Información](#)

Creación estándar Puede definir todas las opciones de configuración, incluidas las de disponibilidad, seguridad, copias de seguridad y mantenimiento.

Creación sencilla Utilice las configuraciones recomendadas. Algunas opciones de configuración se pueden cambiar después de crear la base de datos.

Opciones del motor [Información](#)

Tipo de motor [Información](#)

Aurora (MySQL Compatible)

Aurora (PostgreSQL Compatible)

MySQL

MariaDB

PostgreSQL

Oracle

MySQL

MySQL es la base de datos de código abierto más popular del mundo. MySQL en RDS ofrece las completas características de la edición comunitaria de MySQL con la flexibilidad necesaria para escalar fácilmente los recursos de computación o la capacidad de almacenamiento de la base de datos.

- Admite un tamaño de base de datos máximo de 64 TiB.
- Admite las clases de instancias de uso general, optimizadas para memoria y de rendimiento ampliable.
- Admite las copias de seguridad automatizadas y la recuperación a un momento dado.
- Admite hasta 15 réplicas de lectura por instancia, dentro de una única región, o 5 réplicas de lectura entre regiones.

CloudShell Comentarios © 2024, Amazon Web Services, Inc. o sus filiales. Privacidad Términos Preferencias de cookies

Búsqueda ESP LAA 18:59 9/04/2024

RDS | us-east-2

us-east-2.console.aws.amazon.com/rds/home?region=us-east-2#launch-dbinstance:

YouTube Maps Gmail Azure - Sign up Course overview | G..

Todos los marcadores

AWS Servicios Buscar [Alt+S]

Versión del motor [Información](#)

Vea las versiones de motor que admiten las siguientes características de base de datos.

Mostrar las versiones compatibles con el clúster de base de datos multi-AZ [Información](#)

Crear un clúster de base de datos multi-AZ con una instancia de base de datos principal y dos instancias de base de datos en espera que se puedan leer. Los clúusters de base de datos multi-AZ ofrecen una latencia de confirmación de transacciones hasta dos veces más rápida y comutación por error automática en menos de 35 segundos.

Mostrar versiones compatibles con las escrituras optimizadas de Amazon RDS [Información](#)

Las escrituras optimizadas de Amazon RDS mejoran el rendimiento de escritura hasta 2 veces sin costo adicional.

Versión del motor

MySQL 8.0.35

MySQL

MySQL es la base de datos de código abierto más popular del mundo. MySQL en RDS ofrece las completas características de la edición comunitaria de MySQL con la flexibilidad necesaria para escalar fácilmente los recursos de computación o la capacidad de almacenamiento de la base de datos.

- Admite un tamaño de base de datos máximo de 64 TiB.
- Admite las clases de instancias de uso general, optimizadas para memoria y de rendimiento ampliable.
- Admite las copias de seguridad automatizadas y la recuperación a un momento dado.
- Admite hasta 15 réplicas de lectura por instancia, dentro de una única región, o 5 réplicas de lectura entre regiones.

CloudShell Comentarios © 2024, Amazon Web Services, Inc. o sus filiales. Privacidad Términos Preferencias de cookies

Búsqueda ESP LAA 18:59 9/04/2024

Disponibilidad y durabilidad

Opciones de implementación [Información](#)
Las siguientes opciones de implementación están limitadas a las compatibles con el motor que ha seleccionado anteriormente.

- Clúster de base de datos multi-AZ
Crea un clúster de base de datos con una instancia de base de datos primaria y dos instancias de base de datos en espera con capacidad de lectura, con cada instancia de base de datos en una zona de disponibilidad (AZ) diferente. Proporciona alta disponibilidad, redundancia de datos y aumenta la capacidad de incluir cargas de trabajo de lectura.
- Instancia de base de datos Multi-AZ (no compatible con la instantánea de clúster de base de datos Multi-AZ)
Crea una instancia de base de datos primaria y una instancia de base de datos en espera en una zona de disponibilidad diferente. Proporciona alta disponibilidad y redundancia de datos, pero la instancia de base de datos en espera no admite conexiones para trabajo de lectura.
- Instancia de base de datos única (no compatible con la instantánea de clúster de base de datos Multi-AZ)
Crea una sola instancia de base de datos sin instancias de base de datos en espera.

Configuración

Identificador de instancias de bases de datos [Información](#)
Escriba un nombre para la instancia de base de datos. El nombre debe ser único en relación con todas las instancias de base de datos pertenecientes a su cuenta de AWS en la región de AWS actual.

El identificador de la instancia de base de datos no distingue entre mayúsculas y minúsculas, pero se almacena con todas las letras en minúsculas (como en "minstantáneidad"). Restricciones: de 1 a 60 caracteres alfanuméricos o guiones. El primer carácter debe ser una letra. No puede contener dos guiones consecutivos. No puede terminar con un guion.

MySQL

MySQL es la base de datos de código abierto más popular del mundo. MySQL en RDS ofrece las completas características de la edición comunitaria de MySQL con la flexibilidad necesaria para escalar fácilmente los recursos de computación o la capacidad de almacenamiento de la base de datos.

- Admite un tamaño de base de datos máximo de 64 TiB.
- Admite las clases de instancias de uso general, optimizadas para memoria y de rendimiento ampliable.
- Admite las copias de seguridad automatizadas y la recuperación a un momento dado.
- Admite hasta 15 réplicas de lectura por instancia, dentro de una única región, o 5 réplicas de lectura entre regiones.

Configuración

Identificador de instancias de bases de datos [Información](#)
Escriba un nombre para la instancia de base de datos. El nombre debe ser único en relación con todas las instancias de base de datos pertenecientes a su cuenta de AWS en la región de AWS actual.

El identificador de la instancia de base de datos no distingue entre mayúsculas y minúsculas, pero se almacena con todas las letras en minúsculas (como en "minstantáneidad"). Restricciones: de 1 a 60 caracteres alfanuméricos o guiones. El primer carácter debe ser una letra. No puede contener dos guiones consecutivos. No puede terminar con un guion.

Configuración de credenciales

Nombre de usuario maestro [Información](#)
Escriba un ID de inicio de sesión para el usuario maestro de la instancia de base de datos.

1 a 16 caracteres alfanuméricos. El primer carácter debe ser una letra.

Credentials management
You can use AWS Secrets Manager or manage your master user credentials.

Managed in AWS Secrets Manager - most secure
RDS generates a password for you and manages it throughout its lifecycle using AWS Secrets Manager.

Self managed
Create your own password or have RDS create a password that you manage.

Auto generate password

Amazon RDS automatically creates a password for you, or you can enter your own password.

MySQL

MySQL es la base de datos de código abierto más popular del mundo. MySQL en RDS ofrece las completas características de la edición comunitaria de MySQL con la flexibilidad necesaria para escalar fácilmente los recursos de computación o la capacidad de almacenamiento de la base de datos.

- Admite un tamaño de base de datos máximo de 64 TiB.
- Admite las clases de instancias de uso general, optimizadas para memoria y de rendimiento ampliable.
- Admite las copias de seguridad automatizadas y la recuperación a un momento dado.
- Admite hasta 15 réplicas de lectura por instancia, dentro de una única región, o 5 réplicas de lectura entre regiones.

RDS | us-east-2

us-east-2.console.aws.amazon.com/rds/home?region=us-east-2#launch-dbinstance:

YouTube Maps Gmail Azure - Sign up Course overview | G...

Servicios Buscar [Alt+S]

Credentials management

You can use AWS Secrets Manager or manage your master user credentials.

Managed in AWS Secrets Manager - most secure RDS generates a password for you and manages it throughout its lifecycle using AWS Secrets Manager.

Self managed Create your own password or have RDS create a password that you manage.

Auto generate password Amazon RDS can generate a password for you, or you can specify your own password.

Master password [Información](#)

Confirm master password [Información](#)

Configuración de la instancia

Las opciones de configuración de la instancia de base de datos que aparecen a continuación están limitadas a las que admite el motor que ha seleccionado anteriormente.

Clase de instancia de base de datos [Información](#)

▼ Ocultar filtros

Mostrar las clases de instancia que admiten las escrituras optimizadas de Amazon RDS [Información](#) Las escrituras optimizadas de Amazon RDS mejoran el rendimiento de escritura hasta 2 veces sin costo adicional.

Incluir clases de generación anterior

MySQL

MySQL es la base de datos de código abierto más popular del mundo. MySQL en RDS ofrece las completas características de la edición comunitaria de MySQL con la flexibilidad necesaria para escalar fácilmente los recursos de computación o la capacidad de almacenamiento de la base de datos.

- Admite un tamaño de base de datos máximo de 64 TiB.
- Admite las clases de instancias de uso general, optimizadas para memoria y de rendimiento ampliable.
- Admite las copias de seguridad automatizadas y la recuperación a un momento dado.
- Admite hasta 15 réplicas de lectura por instancia, dentro de una única región, o 5 réplicas de lectura entre regiones.

CloudShell Comentarios

Búsqueda

© 2024, Amazon Web Services, Inc. o sus filiales. Privacidad Términos Preferencias de cookies

ESP LAA 19:03 9/04/2024

RDS | us-east-2

us-east-2.console.aws.amazon.com/rds/home?region=us-east-2#launch-dbinstance:

YouTube Maps Gmail Azure - Sign up Course overview | G...

Servicios Buscar [Alt+S]

▼ Ocultar filtros

Mostrar las clases de instancia que admiten las escrituras optimizadas de Amazon RDS [Información](#) Las escrituras optimizadas de Amazon RDS mejoran el rendimiento de escritura hasta 2 veces sin costo adicional.

Incluir clases de generación anterior

Clases estándar (incluye clases m)

Clases optimizadas para memoria (incluye clases r y x)

Clases con ráfagas (incluye clases t)

db.t3.micro
2 vCPUs 1 GiB RAM Red: 2085 Mbps

Almacenamiento

Tipo de almacenamiento [Información](#) Los volúmenes de almacenamiento SSD de IOPS aprovisionadas (io2) ya están disponibles.

SSD de uso general (gp2)
Rendimiento de referencia determinado por el tamaño del volumen

Almacenamiento asignado [Información](#)

20 GiB

El valor mínimo es 20 GiB y el valor máximo es 5144 GiB

MySQL

MySQL es la base de datos de código abierto más popular del mundo. MySQL en RDS ofrece las completas características de la edición comunitaria de MySQL con la flexibilidad necesaria para escalar fácilmente los recursos de computación o la capacidad de almacenamiento de la base de datos.

- Admite un tamaño de base de datos máximo de 64 TiB.
- Admite las clases de instancias de uso general, optimizadas para memoria y de rendimiento ampliable.
- Admite las copias de seguridad automatizadas y la recuperación a un momento dado.
- Admite hasta 15 réplicas de lectura por instancia, dentro de una única región, o 5 réplicas de lectura entre regiones.

CloudShell Comentarios

Búsqueda

© 2024, Amazon Web Services, Inc. o sus filiales. Privacidad Términos Preferencias de cookies

ESP LAA 19:03 9/04/2024

RDS | us-east-2

us-east-2.console.aws.amazon.com/rds/home?region=us-east-2#launch-dbinstance:

YouTube Maps Gmail Azure - Sign up Course overview | G...

Servicios Buscar [Alt+S]

MySQL

Conectividad

Recurso de computación

No se conecta a un recurso informático EC2

Conectarse a un recurso informático de EC2

Nube privada virtual (VPC)

Default VPC (vpc-054cc0a8c5303a4a7)

Gráfico de subredes

Acceso público

Sí

CloudShell Comentarios

Búsqueda

© 2024, Amazon Web Services, Inc. o sus filiales. Privacidad Términos Preferencias de cookies

19:07 9/04/2024

MySQL es la base de datos de código abierto más popular del mundo. MySQL en RDS ofrece las completas características de la edición comunitaria de MySQL con la flexibilidad necesaria para escalar fácilmente los recursos de computación o la capacidad de almacenamiento de la base de datos.

- Admite un tamaño de base de datos máximo de 64 TiB.
- Admite las clases de instancias de uso general, optimizadas para memoria y de rendimiento ampliable.
- Admite las copias de seguridad automatizadas y la recuperación a un momento dado.
- Admite hasta 15 réplicas de lectura por instancia, dentro de una única región, o 5 réplicas de lectura entre regiones.

RDS | us-east-2

us-east-2.console.aws.amazon.com/rds/home?region=us-east-2#launch-dbinstance:

YouTube Maps Gmail Azure - Sign up Course overview | G...

Servicios Buscar [Alt+S]

MySQL

Proxy de RDS

Creación de un proxy de RDS

Entidad de certificación - opcional

Tags

Puerto de la base de datos

3306

CloudShell Comentarios

Búsqueda

© 2024, Amazon Web Services, Inc. o sus filiales. Privacidad Términos Preferencias de cookies

19:09 9/04/2024

MySQL es la base de datos de código abierto más popular del mundo. MySQL en RDS ofrece las completas características de la edición comunitaria de MySQL con la flexibilidad necesaria para escalar fácilmente los recursos de computación o la capacidad de almacenamiento de la base de datos.

- Admite un tamaño de base de datos máximo de 64 TiB.
- Admite las clases de instancias de uso general, optimizadas para memoria y de rendimiento ampliable.
- Admite las copias de seguridad automatizadas y la recuperación a un momento dado.
- Admite hasta 15 réplicas de lectura por instancia, dentro de una única región, o 5 réplicas de lectura entre regiones.

The screenshot shows the AWS RDS MySQL configuration page. On the left, there are sections for 'Tags' (empty), 'Autenticación de bases de datos' (Authenticación con contraseña selected), and 'Supervisión' (Activar la monitorización mejorada checked). On the right, a sidebar titled 'MySQL' provides a brief overview and a bulleted list of features.

Tags
A tag consists of a case-sensitive key-value pair.
No hay etiquetas asociadas al recurso.
Agregar nueva etiqueta
Puede agregar hasta 50 etiquetas más.

Autenticación de bases de datos

Opciones de autenticación de bases de datos [Información](#)

Autenticación con contraseña
Se autentica con las contraseñas de las bases de datos.

Autenticación de bases de datos con contraseña e IAM
Se autentica con las credenciales de usuario y la contraseña de las bases de datos a través de usuarios y roles de AWS IAM.

Autenticación Kerberos y con contraseña
Elija un directorio en el que deseas permitir que los usuarios autorizados se autentiquen en esta instancia de base de datos a través de la autenticación Kerberos.

Supervisión

Activar la monitorización mejorada

MySQL

MySQL es la base de datos de código abierto más popular del mundo. MySQL en RDS ofrece las completas características de la edición comunitaria de MySQL con la flexibilidad necesaria para escalar fácilmente los recursos de computación o la capacidad de almacenamiento de la base de datos.

- Admite un tamaño de base de datos máximo de 64 TiB.
- Admite las clases de instancias de uso general, optimizadas para memoria y de rendimiento ampliable.
- Admite las copias de seguridad automatizadas y la recuperación a un momento dado.
- Admite hasta 15 réplicas de lectura por instancia, dentro de una única región, o 5 réplicas de lectura entre regiones.

The screenshot shows the AWS RDS MySQL configuration page with the 'Configuración adicional' section expanded. It includes fields for 'Nombre de base de datos inicial' (projectofinal), 'Grupo de parámetros de base de datos' (default.mysql8.0), and 'Grupo de opciones' (default:mysql-8-0). The 'Copia de seguridad' section has a checked checkbox for 'Habilitar las copias de seguridad automatizadas'. A note at the bottom states: 'Tenga en cuenta que, actualmente, las copias de seguridad automáticas son solo compatibles con el motor de almacenamiento InnoDB. Si está usando MyISAM, consulte los detalles [aqui](#)'.

Configuración adicional
Opciones de base de datos, cifrado activado, copia de seguridad activado, retroceder desactivado, mantenimiento, CloudWatch Logs, eliminar protección desactivado.

Opciones de base de datos

Nombre de base de datos inicial [Información](#)
projectofinal
Si no especifica un nombre de base de datos, Amazon RDS no crea una base de datos.

Grupo de parámetros de base de datos [Información](#)
default.mysql8.0

Grupo de opciones [Información](#)
default:mysql-8-0

Copia de seguridad

Habilitar las copias de seguridad automatizadas.
Crea una instantánea de un momento dado de su base de datos

MySQL

MySQL es la base de datos de código abierto más popular del mundo. MySQL en RDS ofrece las completas características de la edición comunitaria de MySQL con la flexibilidad necesaria para escalar fácilmente los recursos de computación o la capacidad de almacenamiento de la base de datos.

- Admite un tamaño de base de datos máximo de 64 TiB.
- Admite las clases de instancias de uso general, optimizadas para memoria y de rendimiento ampliable.
- Admite las copias de seguridad automatizadas y la recuperación a un momento dado.
- Admite hasta 15 réplicas de lectura por instancia, dentro de una única región, o 5 réplicas de lectura entre regiones.

The screenshot shows the AWS RDS console for the 'us-east-2' region. A modal window titled 'MySQL' is open, providing information about the service. The main configuration page includes sections for 'Copia de seguridad' (Backup), 'Cifrado' (Encryption) with a checked checkbox for 'Habilitar el cifrado' (Enable encryption), and 'Clave de AWS KMS' (AWS KMS Key) set to '(default) aws/rds'. Below these are sections for 'Cuenta' (Account) and 'ID de clave de KMS' (KMS Key ID). The 'Exportaciones de registros' (Log exports) section lists several log types: 'Registro de auditoría' (Audit log), 'Registro de errores' (Error log), 'Registro general' (General log), and 'Registro de consultas lentas' (Slow query log), all of which are unchecked. The right side of the modal contains a detailed description of MySQL and a bulleted list of its features.

The screenshot shows the DBBeaver application interface. On the left, a 'Connection Settings' dialog for MySQL is open, showing the 'Connection Settings' tab. It displays the 'Server' section with 'Connect by' set to 'Host' (radio button selected), 'URL' as 'jdbc:mysql://database-proyectorfinal.cz20sgueugi2.us-east-2.rds.amazonaws.com:3306/proyectorfinal', 'Server Host' as 'database-proyectorfinal.cz20sgueugi2.us-east-2.rds.amazonaws.com', 'Port' as '3306', and 'Database' as 'proyectorfinal'. The 'Authentication (Database Native)' section shows 'Nombre de usuario:' as 'admin_mysql' and 'Contraseña:' as a masked password. The 'Advanced' section includes 'Server Time Zone' set to 'Auto-detect' and 'Local Client' set to 'MySQL Binaries'. At the bottom of the dialog are buttons for 'Probar conexión...' (Test connection...), 'Anterior' (Previous), 'Siguiente' (Next), 'Finalizar' (Finish), and 'Cancelar' (Cancel). To the right of the dialog, a 'Connection test' window is open, showing a green status bar with 'Conectado (2103 ms)' and 'Aceptar' (Accept) and 'Detalles' (Details) buttons. The bottom of the screen shows the Windows taskbar with various pinned icons and the date/time as '10/04/2024 21:48'.

```
-- Table structure for table `weather`  
DROP TABLE IF EXISTS `weather`;  
CREATE TABLE `weather` (  
  `origin_id` int NOT NULL AUTO_INCREMENT,  
  `origin` varchar(10) NOT NULL,  
  `year` int NOT NULL,  
  `month` int NOT NULL,  
  `day` int NOT NULL,  
  `hour` int NOT NULL,  
  `temp` decimal(5,2) NOT NULL DEFAULT '19.99',  
  PRIMARY KEY(`origin_id`),  
  KEY `ix_weather` (`origin`)  
) ENGINE=InnoDB AUTO_INCREMENT=17 DEFAULT CHARSET=utf8mb4;
```

- El procesamiento puede realizarlo en un maquina local o en una instancia de EC2 corriendo Python.
- El Notebook debe contener detalles sobre los procesos necesarios para responder las preguntas de negocio planteadas.
- Notar que no puede usar SQL para hacer la construcción de ninguna estructura salvo para leer de tablas almacenadas en las bases de datos es decir SELECT * FROM tabla. Cualquier otra necesidad de procesamiento deberá hacerla en Python con pandas y librerías de procesamiento de información.

SELECT * FROM flights

```
SELECT * FROM flights
```

CARGA DE DATOS DE AIRLINES:

The screenshot shows a Jupyter Notebook window titled 'lab (5) - JupyterLab'. The code cell [17] contains the command `airlines_data = pd.read_csv('../data/airlines.csv')` followed by `airlines_data.head()`. The output displays the first five rows of the 'airlines' dataset, which includes columns 'carrier' and 'name'. The code cell [18] contains the command `airlines_data.to_sql('airlines', mysql_driver, index=False, if_exists='append')` and its output shows the value '16', indicating 16 rows were inserted.

```
[17]: airlines_data = pd.read_csv('../data/airlines.csv')
airlines_data.head()

[17]:
   carrier      name
0      9E Endeavor Air Inc.
1      AA American Airlines Inc.
2      AS Alaska Airlines Inc.
3      B6 JetBlue Airways
4      DL Delta Air Lines Inc.

[18]: airlines_data.to_sql('airlines', mysql_driver, index=False, if_exists='append')
[18]: 16
```

`SELECT * FROM airlines`

The screenshot shows the DBBeaver interface. In the SQL tab, a script named '21.sql' is running, showing the creation of the 'airlines' table. The table has columns 'carrier_id' (primary key), 'carrier' (varchar(50)), and 'name' (varchar(50)). The engine is set to InnoDB with AUTO_INCREMENT=606 and DEFAULT CHARSET=utf8mb4. A message in the 'Salida' (Output) panel indicates that the table was created successfully. In the 'airlines 1' tab, a preview of the table data is shown, listing 16 rows from carrier_id 1 to 612, with names like Endeavor Air Inc., American Airline, Alaska Airlines, JetBlue Airways, Delta Air Lines, ExpressJet Airlines, and Frontier Airlines.

```
-- Table structure for table 'airlines'
DROP TABLE IF EXISTS 'airlines';
CREATE TABLE 'airlines' (
  'carrier_id' int NOT NULL AUTO_INCREMENT,
  'carrier' varchar(50) NOT NULL,
  'name' varchar(50) NOT NULL,
  PRIMARY KEY ('carrier_id'),
  KEY 'ix_airlines' ('carrier')
) ENGINE=InnoDB AUTO_INCREMENT=606 DEFAULT CHARSET=utf8mb4;

SELECT * FROM airlines;
```

carrier_id	carrier	name
1	9E	Endeavor Air Inc.
2	AA	American Airline
3	AS	Alaska Airlines
4	B6	JetBlue Airways
5	DL	Delta Air Lines
6	EV	ExpressJet Airlines
7	F9	Frontier Airlines

Carga de datos de airports y planes

lab (5) - JupyterLab

localhost:8889/lab# Cargamos el archivo airlines.csv

File Eds View Run Kernel Tabs Settings Help

Filter files by name

/ Desktop / Laboratorios Python /

- Name Last Modified
- data 3 days ago
- config.cfg 8 hours ago
- Lec_9_A.ipynb 5 days ago
- nuevo_fliq... 2 days ago
- ProyectoFin... a day ago
- sakila-db_2... 6 days ago
- untitled.txt 18 hours ago

carrier name

	carrier	name
0	9E	Endeavor Air Inc.
1	AA	American Airlines Inc.
2	AS	Alaska Airlines Inc.
3	B6	JetBlue Airways
4	DL	Delta Air Lines Inc.

[17]: carrier name

[18]: airlines_data.to_sql('airlines', mysql_driver, index=False, if_exists='append')

[18]: 16

Cargamos el archivo airports.csv

[19]: airports_data = pd.read_csv('./data/airports.csv')
airports_data.head()

faa	name	lat	lon	alt	tz	dst	tzone
04G	Lansdowne Airport	41.130472	-80.619583	1044	-5	A	America/New_York
06A	Moton Field Municipal Airport	32.460572	-85.680028	264	-6	A	America/Chicago
06C	Schaumburg Regional	41.989341	-88.101243	801	-6	A	America/Chicago
06N	Randall Airport	41.431912	-74.391561	523	-5	A	America/New_York
09J	Jekyll Island Airport	31.074472	-81.427778	11	-5	A	America/New_York

Simple 0 2 Python 3 (ipykernel) | idle Mode: Edit L1 Col 1 ProyectoFinal.ipynb 1 ESP LAA 07:09 14/04/2024

DBBeaver 24.0.2 - <proyectofinal> 21.sql

Archivo Editar Navegar Buscar Editor SQL Base de Datos Ventana Ayuda

Tareas de base de datos - General Navegador de Bases de Datos

Ingrese parte del nombre de un objeto aquí

proyectofinal - database-projectofinal.c20geuegi2.us-east-2.rds.amazonaws.com:3306

Databases

- projectofinal
 - Tables
 - airlines
 - airports
 - flights
 - Views
 - Indexes
 - Procedures

projectofinal <proyecto> 21.sql <proyecto> Script-4

```

`name` varchar(50) NOT NULL,
`lat` decimal(5,2) NOT NULL DEFAULT '19.99',
`lon` decimal(5,2) NOT NULL DEFAULT '19.99',
`alt` int NOT NULL,
`tz` int NOT NULL,
`dst` int(10) NOT NULL,
`tzone` varchar(50) NOT NULL,
PRIMARY KEY (`airports_id`),
KEY `ix_airports` (`faa`)
) ENGINE=InnoDB AUTO_INCREMENT=17 DEFAULT CHARSET=utf8mb4;

SELECT * FROM airports;

```

airports 1

Enter a SQL expression to filter results (use Ctrl+Space)

airports_id	faa	name	lat	lon	alt	tz	dst	tzone

Record Grid Text

Renovar Save Cancel Exportar datos ... 200 0 ... No data- 0.076s, on 2024-04-14 at 07:10:25 CST es_GT Editable Inserción inteligente 62:24:1827 Sel: 0 | 0 ... ESP LAA 07:10 14/04/2024

lab (5) - JupyterLab

localhost:8889/lab#Cargamos el archivo airports.csv

File Eds View Run Kernel Tabs Settings Help

ProyectoFinal.ipynb Lec_9_A.ipynb nuevo_flights.txt config.cfg Python 3 (ipykernel)

Filter files by name

Name Last Modified

- data 3 days ago
- config.cfg 8 hours ago
- Lec_9_A.ipynb 5 days ago
- nuevo_flights.txt 2 days ago
- ProyectoFinal.ipynb a minute ago
- sakila-db_2... 6 days ago
- untitled.txt 18 hours ago

3 86 JetBlue Airways

4 DL Delta Air Lines Inc.

```
[18]: airlines_data.to_sql('airlines', mysql_driver, index=False, if_exists='append')
```

```
[18]: 16
```

Cargamos el archivo airports.csv

```
[19]: airports_data = pd.read_csv('./data/airports.csv')
airports_data.head()
```

	faa	name	lat	lon	alt	tz	dst	tzone
0	04G	Lansdowne Airport	41.130472	-80.619583	1044	-5	A	America/New_York
1	06A	Moton Field Municipal Airport	32.460572	-85.680028	264	-6	A	America/Chicago
2	06C	Schaumburg Regional	41.989341	-88.101243	801	-6	A	America/Chicago
3	06N	Randall Airport	41.431912	-74.391561	523	-5	A	America/New_York
4	09J	Jekyll Island Airport	31.074472	-81.427778	11	-5	A	America/New_York

```
[20]: airports_data.to_sql('airports', mysql_driver, index=False, if_exists='append')
```

```
[20]: 1458
```

Simple 0 2 Python 3 (ipykernel) | idle

Búsqueda 07:11 14/04/2024

DBBeaver 24.0.2 - <proyectofinal> 21.sql

Archivo Editor Navegar Buscar Editor SQL Base de Datos Ventana Ayuda

Tareas de base de datos - General Navegador de Bases de Datos

Ingresar parte del nombre de un objeto aquí

proyectofinal - database-proyectofinal.cz20geuegl2.us-east-2.rds.amazonaws.com:3306

- Databases
 - proyectofinal
 - Tables
 - airlines
 - airports
 - flights
 - Views
 - Indexes
 - Procedures

projectofinal <proyectofinal> 21.sql <proyectofinal> Script-4

```
CREATE TABLE `airports` (
  `name` varchar(50) NOT NULL,
  `lat` decimal(5,2) NOT NULL DEFAULT '19.99',
  `lon` decimal(5,2) NOT NULL DEFAULT '19.99',
  `alt` int NOT NULL,
  `tz` int NOT NULL,
  `dst` int(10) NOT NULL,
  `tzone` varchar(50) NOT NULL,
  PRIMARY KEY (`airports_id`),
  KEY `ix_airports` (`faa`)
) ENGINE=InnoDB AUTO_INCREMENT=17 DEFAULT CHARSET=utf8mb4;

SELECT * FROM airports;
```

airports 1

SQL Enter a SQL expression to filter results (use Ctrl+Space)

airports_id	faa	name	lat	lon	alt	tz	dst	tzone
1	04G	Lansdowne Airport	41.13	-80.62	1,044	-5	A	America/New_York
2	18 06A	Moton Field Mu	32.46	-85.68	264	-6	A	America/Chicago
3	19 06C	Schaumburg Rec	41.99	-88.1	801	-6	A	America/Chicago
4	20 06N	Randall Airport	41.43	-74.39	523	-5	A	America/New_York
5	21 09J	Jekyll Island Airp	31.07	-81.43	11	-5	A	America/New_York
6	22 049	Elizabeth Mu	36.37	-82.17	1,593	-5	A	America/New_York
7	23 06G	Williams County	41.47	-84.51	730	-5	A	America/New_York

Renovar Save Cancel Exportar datos ... 200 200+ 200 row(s) fetched - 0.160s (0.008s fetch), on 2024-04-14 at 07:11:06

Record Text

Valor

Renovar Save Cancel Exportar datos ... 200 200+ 200 row(s) fetched - 0.160s (0.008s fetch), on 2024-04-14 at 07:11:06

07:11 14/04/2024

The screenshot displays a dual-monitor setup. The left monitor shows a Jupyter Notebook interface with several tabs open, including 'ProyectoFinal.ipynb' and 'Lec_9_A.ipynb'. The notebook contains Python code for reading CSV files and writing them to MySQL databases. The right monitor shows a DBBeaver database client connected to an Amazon RDS instance. The DBBeaver interface includes a navigation bar, a tree view of the database schema, and a SQL editor window containing the same code as the Jupyter notebook. A 'Salida' (Output) panel is visible on the right side of the DBBeaver window.

DBBeaver 24.0.2 - <proyectofinal> 21.sql

Archivo Editor Navegar Buscar Editor SQL Base de Datos Ventana Ayuda

Tareas de base de datos - General Navegador de Bases de Datos

Ingresé parte del nombre de un objeto aquí

projectofinal - database-projectofinal.cz20geuegi2.us-east-2.rds.amazonaws.com:3306

projectofinal

- Tables
 - airlines
 - airports
 - flights
- Views
- Indexes
- Procedures

projectofinal <proyectofinal> 21.sql <proyectofinal> Script-4

```

CREATE TABLE `planes` (
  `id` int(11) NOT NULL,
  `tailnum` varchar(50) NOT NULL,
  `name` varchar(50) NOT NULL,
  `model` varchar(50) NOT NULL,
  `engines` int(11) NOT NULL,
  `seats` int(11) NOT NULL,
  `speed` varchar(25) NOT NULL,
  `engine` varchar(25) NOT NULL,
  PRIMARY KEY (`id`),
  KEY `ix_planes` (`tailnum`)
) ENGINE=InnoDB AUTO_INCREMENT=606 DEFAULT CHARSET=utf8mb4;

SELECT * FROM planes;

```

Estadísticas 1 planes 2

Record	planes id	tailnum	year	type	manufacturer	model	engines	seats	speed	engine
1	606	N10156	2004	Fixed wing multi engine	EMBRAER	EMB-145XR	2	55	Turbo-fan	
2	607	N102UW	1998	Fixed wing multi engine	AIRBUS INDUSTRIE	A320-214	2	182	Turbo-fan	
3	608	N103US	1999	Fixed wing multi engine	AIRBUS INDUSTRIE	A320-214	2	182	Turbo-fan	
4	609	N104UW	1999	Fixed wing multi engine	AIRBUS INDUSTRIE	A320-214	2	182	Turbo-fan	
5	610	N10575	2002	Fixed wing multi engine	EMBRAER	EMB-145LR	2	55	Turbo-fan	
6	611	N1030W	1999	Fixed wing multi engine	AIRBUS INDUSTRIE	A320-214	2	182	Turbo-fan	
7	612	N107US	1999	Fixed wing multi engine	AIRBUS INDUSTRIE	A320-214	2	182	Turbo-fan	

Salida

```

Unknown table 'projectofinal.airlines'
Unknown table 'projectofinal.airlines'
Unknown table 'projectofinal.airlines'
Unknown table 'projectofinal.airports'
Unknown table 'projectofinal.planes'

```

Valor 606

Búsqueda 07:37 14/04/2024

Carga de datos de weather en csv

lab (5) ~ JupyterLab

localhost:8889/lab# Cargamos el archivo airlines.csv

File Edit View Run Kernel Tabs Settings Help

ProjetoFinalipybnb Lec_9_A.ipynb nuevo_flights.txt config.cfg

Filter files by name

- / Desktop / Laboratorios Python /
- Name Last Modified
- data 3 days ago
- config.cfg 8 hours ago
- Lec_9_A.ipynb 5 days ago
- nuevo_flights.txt 2 days ago
- ProjetoFinalipybnb a day ago
- ProjetoFinalipybnb seconds ago
- sakila-db 6 days ago
- untitled.txt 19 hours ago

```

[22]: planes_data.to_sql('planes', mysql_driver, index=False, if_exists='append')

[22]: 3322

```

Cargamos el archivo weather.csv

```

[23]: weather_data = pd.read_csv('./data/weather.csv')
weather_data.head()

[23]: origin year month day hour temp dewp humid wind_dir wind_speed wind_gust precip pressure visib time_hour
0 EWR 2013 1 1 1 39.02 26.06 59.37 270.0 10.35702 NaN 0.0 1012.0 10.0 2013-01-01T06:00:00Z
1 EWR 2013 1 1 2 39.02 26.96 61.63 250.0 8.05546 NaN 0.0 1012.3 10.0 2013-01-01T07:00:00Z
2 EWR 2013 1 1 3 39.02 28.04 64.43 240.0 11.50780 NaN 0.0 1012.5 10.0 2013-01-01T08:00:00Z
3 EWR 2013 1 1 4 39.92 28.04 62.21 250.0 12.65858 NaN 0.0 1012.2 10.0 2013-01-01T09:00:00Z
4 EWR 2013 1 1 5 39.02 28.04 64.43 260.0 12.65858 NaN 0.0 1011.9 10.0 2013-01-01T10:00:00Z

```

```

[*]: weather_data.to_sql('weather', mysql_driver, index=False, if_exists='append')

```

Simple 0 2 Python 3 (ipykernel) | Busy Mode: Command L 1 Col 78 ProjetoFinalipybnb 1 07:42 14/04/2024

The screenshot shows the DBBeaver interface with the following details:

- Toolbar:** Archivo, Editor, Navegar, Buscar, Editor SQL, Base de Datos, Ventana, Ayuda.
- Database Navigator:** Databases (selected), proyectofinal (selected), Tables, Views, Indexes, Procedures.
- SQL Editor:** Script-4 contains a SELECT query for the 'weather' table.
- Results Grid:** Estadísticas 1 (Statistics 1) shows data for the 'weather' table. The columns are: weather_id, origin, year, month, day, hour, temp, dewp, humid, wind_dir, wind. The data includes rows for EWR airport in 2013.
- Output Panel:** Salida shows a list of unknown tables: 'proyectofinal.airlines', 'proyectofinal.airlines', 'proyectofinal.airports', 'proyectofinal.flights', 'proyectofinal.planes', 'proyectofinal.planes', 'proyectofinal.planes', 'proyectofinal.planes', 'proyectofinal.weather'.

```
--  
-- Table structure for table `flights`  
  
--  
DROP TABLE IF EXISTS `flights`;  
CREATE TABLE `flights` (  
  `flights_id` int NOT NULL AUTO_INCREMENT,  
  `year` int NOT NULL,  
  `month` int NOT NULL,  
  `day` int NOT NULL,  
  `dep_time` int NOT NULL,  
  `sched_dep_time` int NOT NULL,  
  `dep_delay` int NOT NULL,  
  `arr_time` int NOT NULL,  
  `sched_arr_time` int NOT NULL,  
  `arr_delay` int NOT NULL,  
  `carrier` varchar(45) NOT NULL,  
  `flight` int NOT NULL,  
  `tailnum` varchar(45) NOT NULL,  
  `origin` varchar(45) NOT NULL,  
  `dest` varchar(45) NOT NULL,  
  `air_time` int NOT NULL,  
  `distance` int NOT NULL,  
  `hour` int NOT NULL,  
  `minute` int NOT NULL,  
  `time_hour` datetime(3) NOT NULL DEFAULT CURRENT_TIMESTAMP(3),  
  PRIMARY KEY (`flights_id`),  
  KEY `ix_flights` (`year`)  
) ENGINE=InnoDB AUTO_INCREMENT=201 DEFAULT CHARSET=utf8mb4;
```

```
SELECT * FROM flights;
```

```
-- Table structure for table `airlines`
```

```
DROP TABLE IF EXISTS `airlines`;
CREATE TABLE `airlines` (
  `carrier_id` int NOT NULL AUTO_INCREMENT,
  `carrier` varchar(50) NOT NULL,
  `name` varchar(50) NOT NULL,
  PRIMARY KEY (`carrier_id`),
  KEY `ix_airlines` (`carrier`)
) ENGINE=InnoDB AUTO_INCREMENT=606 DEFAULT CHARSET=utf8mb4;
```

```
SELECT * FROM airlines;
```

```
-- Table structure for table `airports`
```

```
DROP TABLE IF EXISTS `airports`;
CREATE TABLE `airports` (
  `airports_id` int NOT NULL AUTO_INCREMENT,
  `faa` varchar(50) NOT NULL,
  `name` varchar(50) NOT NULL,
  `lat` decimal(5,2) NOT NULL DEFAULT '19.99',
  `lon` decimal(5,2) NOT NULL DEFAULT '19.99',
  `alt` int NOT NULL,
  `tz` int NOT NULL,
  `dst` varchar(10) NOT NULL,
  `tzone` varchar(50) NOT NULL,
  PRIMARY KEY (`airports_id`),
  KEY `ix_airpots` (`faa`)
) ENGINE=InnoDB AUTO_INCREMENT=17 DEFAULT CHARSET=utf8mb4;
```

```
SELECT * FROM airports;
```

```
-- Table structure for table `planes`
```

```
DROP TABLE IF EXISTS `planes`;
CREATE TABLE `planes` (
  `planes_id` int NOT NULL AUTO_INCREMENT,
  `tailnum` varchar(10) NOT NULL,
  `year` varchar(10) NOT NULL,
  `type` varchar(50) NOT NULL,
  `manufacturer` varchar(50) NOT NULL,
  `model` varchar(50) NOT NULL,
  `engines` int NOT NULL,
  `seats` int NOT NULL,
  `speed` varchar(25) NOT NULL,
  `engine` varchar(25) NOT NULL,
  PRIMARY KEY (`planes_id`),
  KEY `ix_planes` (`tailnum`)
) ENGINE=InnoDB AUTO_INCREMENT=606 DEFAULT CHARSET=utf8mb4;
```

```
SELECT * FROM planes;
```

```
-- Table structure for table `weather`
```

```
DROP TABLE IF EXISTS `weather`;
CREATE TABLE `weather` (
  `weather_id` int NOT NULL AUTO_INCREMENT,
  `origin` varchar(10) NOT NULL,
  `year` int NOT NULL,
  `month` int NOT NULL,
  `day` int NOT NULL,
  `hour` int NOT NULL,
```

```

`temp` decimal(5,2) NOT NULL DEFAULT '19.99',
`dewp` decimal(5,2) NOT NULL DEFAULT '19.99',
`humid` decimal(5,2) NOT NULL DEFAULT '19.99',
`wind_dir` varchar(25) NOT NULL,
`wind_speed` decimal(5,2) NOT NULL DEFAULT '19.99',
`wind_gust` varchar(25) NOT NULL,
`precip` int NOT NULL,
`pressure` decimal(5,2) NOT NULL DEFAULT '19.99',
`visib` int NOT NULL,
`time_hour` datetime(3) NOT NULL DEFAULT CURRENT_TIMESTAMP(3),
PRIMARY KEY (`weather_id`),
KEY `ix_weather` (`origin`)
) ENGINE=InnoDB AUTO_INCREMENT=17 DEFAULT CHARSET=utf8mb4;

```

ETL y Analytics: Utilizando Markdown, numpy, pandas, matplotlib y seaborn, deberá elaborar un notebook con las respuestas a la preguntas planteadas de forma consistente y lo más profundo que sea posible.

Para este caso deberá mostrar el código necesario para resolver cada pregunta planeada. Adicionalmente deberá responder las siguientes preguntas de negocio utilizando tablas y gráficas:

1. ¿En qué país y qué avión se encuentra entre el 85% y el 70% de la cantidad de aterrizajes?
2. Proporcione el nombre del país y el nombre del avión. 2. ¿Indique cuál es el aeropuerto con la temperatura más alta registrada en los datos?
3. ¿Cuál es la aerolínea con la menor cantidad de vuelos registrados, indique cuantos vuelos, el código de la aerolínea, el nombre completo?
4. Indique la media, median, mínimo, máximo y desviación estándar de las millas recorridas por cada avión, debe mostrar el nombre del avión y la información estadística en columnas adicionales
5. Muestre un cubo de información incluyendo la información de todas las tablas proporcionadas.
6. Indique la cantidad de filas y columnas b. Indique cuantas y cuales son las variables categóricas, continuas, discretas y de fecha y hora. c. Muestre una gráfica i. de barras para la cantidad de las variables categóricas y discretas. ii. De densidad para las variables continuas. iii. Serie de tiempo con el conteo de apariciones para las de fecha y hora.

DETALLES TECNICOS: A continuación se describen los detalles técnicos mínimos que su proyecto debe cumplir:

- Para desarrollar el sistema transaccional podrá utilizar cualquier gestor de base de datos SQL que esté disponible en RDS.
- El procesamiento puede realizarlo en un maquina local o en una instancia de EC2 corriendo Python.
- El Notebook debe contener detalles sobre los procesos necesarios para responder las preguntas de negocio planteadas.
- Notar que no puede usar SQL para hacer la construcción de ninguna estructura salvo para leer de tablas almacenadas en las bases de datos es decir `SELECT * FROM tabla`. Cualquier otra necesidad de procesamiento deberá hacerla en Python con pandas y librerías de procesamiento de información

ENTREGA: Como entrega deberá publicar todos los archivos utilizados por medio de un link de Git, incluyendo la presentación utilizada en el video, los notebooks utilizados, los y los archivos adicionales que requiera. Adicionalmente deberá hacer un video de 5 a 7 minutos máximo donde explique todos los pasos que realizó para desarrollar su proyecto, es decir describir todos los elementos de su proyecto.

Enlace de Github

<https://github.com/github20124/Proyecto-Final-Python>

Enlace de video

https://drive.google.com/file/d/1NNuCXL1Spn5SSXayHYx4-OZzyCMa_9rQ/view?usp=sharing

Enlace de diapositivas

<https://docs.google.com/presentation/d/19VNbYjz3L5AvypxIQ3tNHnhK5p80NSQj/edit?usp=sharing&oid=104936982114351849800&rtpof=true&sd=true>