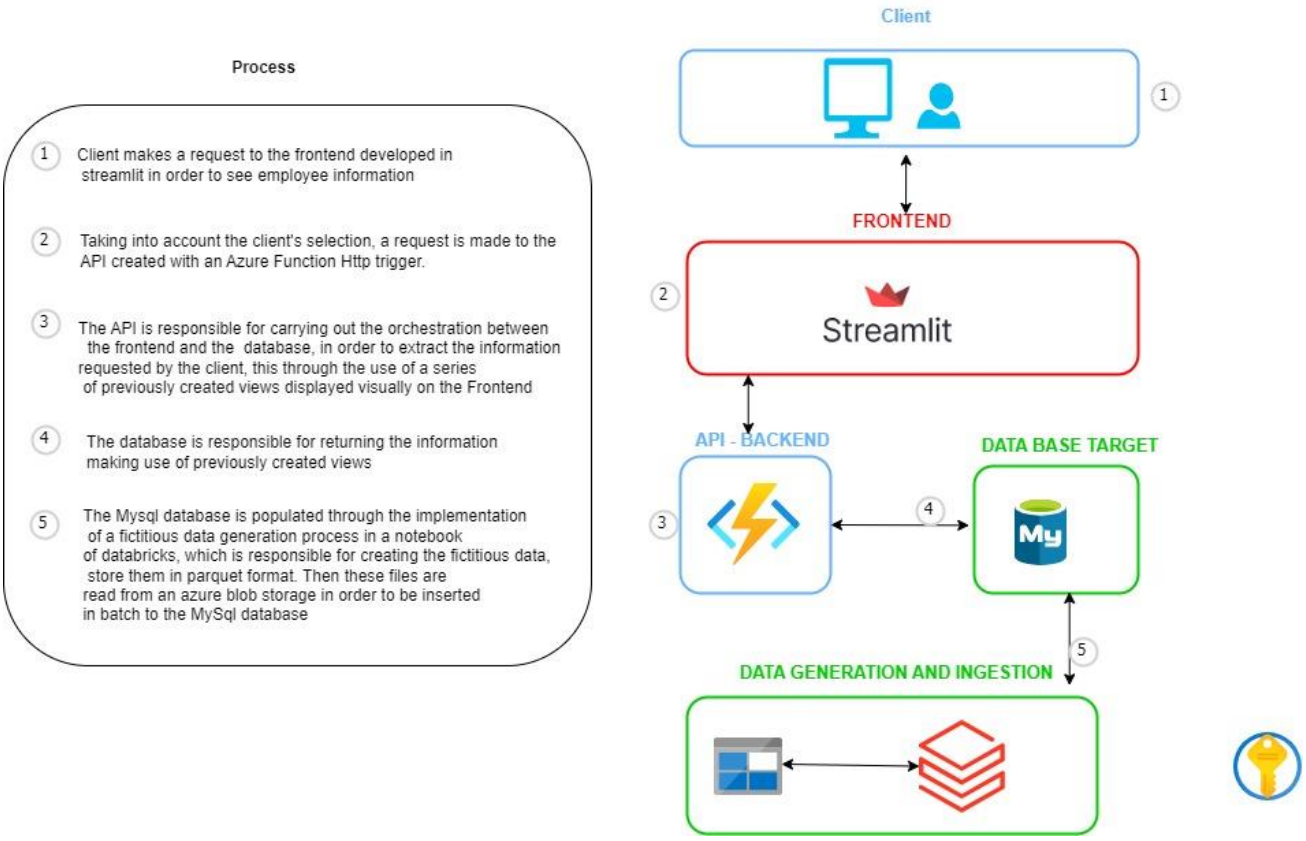
Arquitectura de la Solución Planteada



DESAFIO #1:

Construya un script que genere de forma automática los datos de: departamentos, puestos de trabajo, y empleados.

Solución:

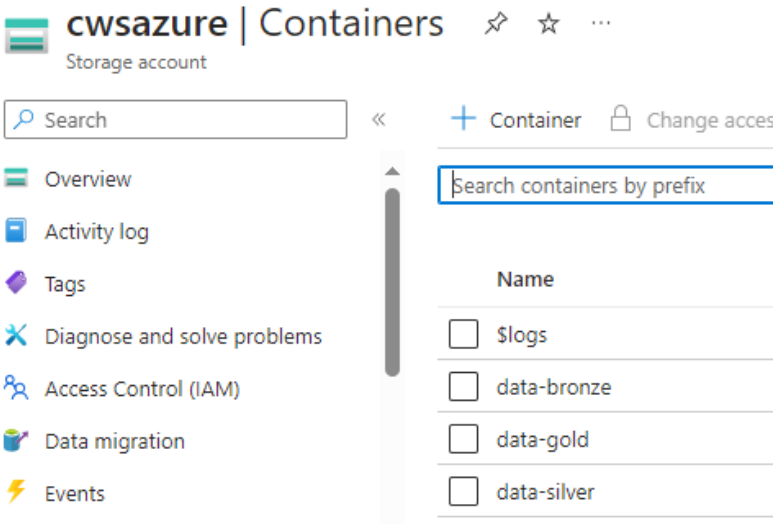
El presente desafío se soluciona en un notebook de Databricks llamado “TestMVM”, este notebook es creado en Databricks comunnity edition.

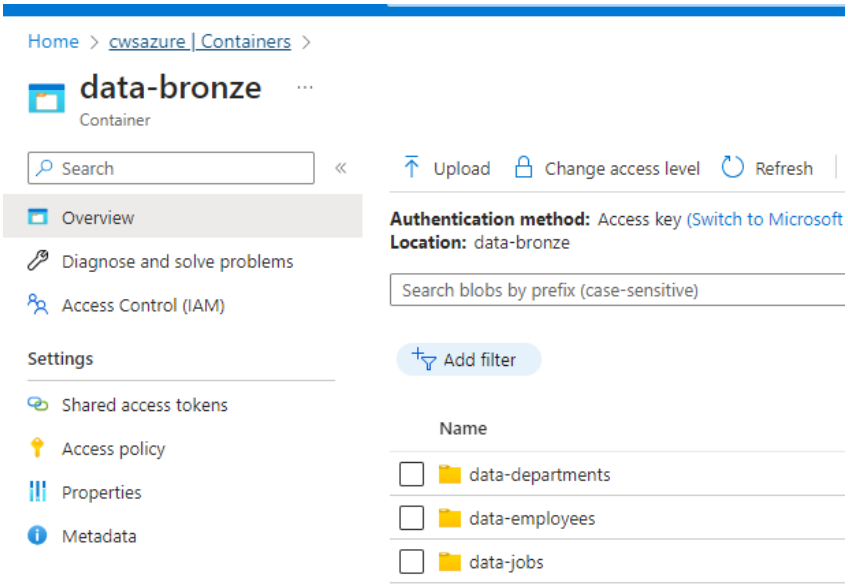
DESAFIO #2

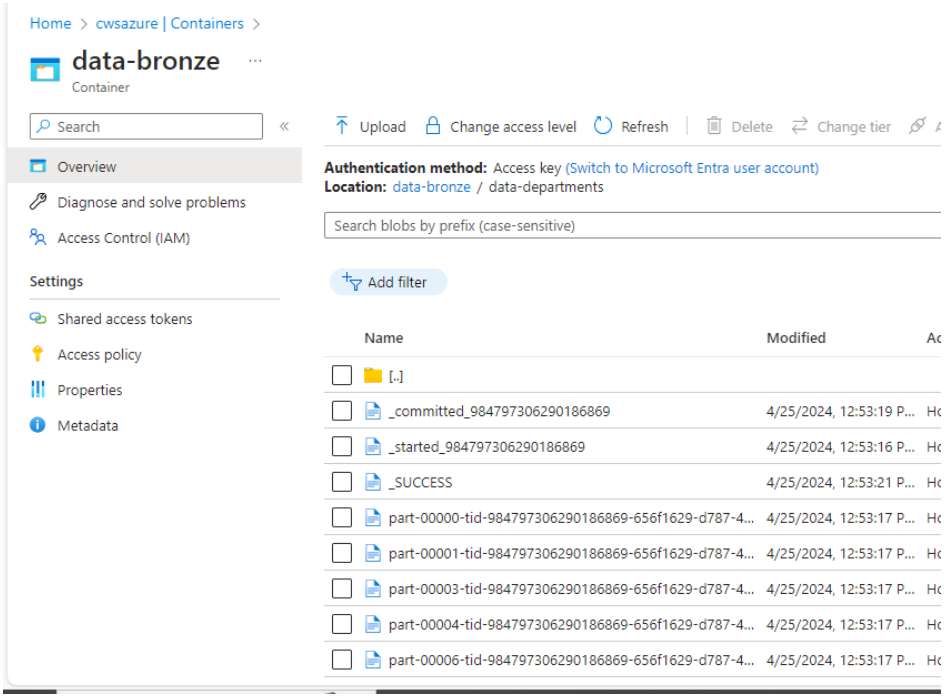
Guarde los datos simulados en archivos con formato CSV/Parquet. Explique el porqué de la escogencia del formato. No descarte usar la capa gratuita de algún servicio de almacenamiento tipo cloud, será considerado un plus.

Solución :

Los datos generados son almacenados en un Azure Blob storage, se crea la gestión de capas en blob storage de bronze, silver y gold.







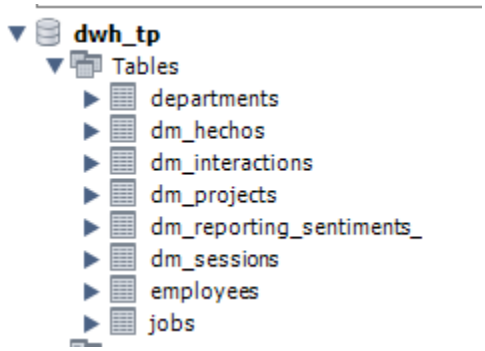
Los datos son almacenados en la capa bronze, en formato parquet ya que este formato es optimizado para el almacenamiento de datos tabulares, este formato realiza una compresión de los datos de forma eficiente permitiendo tener un menor tamaño en comparación a otros formatos. Permitiendo asi tener una reducción en los costos de almacenamiento en el blob storage.

DESAFIO #3:

Implemente un proceso batch para migrar los datos a una base de datos SQL/NoSQL, o si lo desea, a un Datawarehouse o bucket analítico de un Datalake. No descarte usar la capa gratuita de algún servicio de almacenamiento tipo cloud, será considerado un plus.

Solución:

Se realiza una lectura de los archivos tipo parquet desde un notebook en databricks, el cual lee los archivos y los almacena en batch a una base de datos Mysql, en la cual se crean 3 tablas (employees, Jobs, departments) con el fin de almacenar la información que proviene del blob storage.




```
3 •  SELECT * FROM dwh_tp.departments;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

employee_id	department_id	job_id	first_name	last_name	email	phone_number	contract_date	salary
1	5	3	Brandon	Miller	ashleywilcox@example.org	361.292.3713x6066	2020-07-08	54413
2	1	1	John	Osborne	jimenezbrandon@example.com	752-398-5954x03932	2020-09-16	56586
3	2	3	Micheal	Brown	amypham@example.com	(342)281-4004x61433	2021-01-21	79661
4	3	4	Dean	Smith	hamiltontom@example.com	365-210-9127	2023-05-21	42101
5	4	2	Mark	Holloway	ycampbell@example.net	301-353-2266	2024-04-11	63936
6	4	5	Adam	Anthony	andersonlucas@example.org	(830)867-1970x8598	2021-06-23	40616
7	2	5	Amanda	Mcdonald	michaelmunoz@example.net	(216)746-8408x70241	2023-08-03	63136
8	1	4	Alan	Kennedy	william25@example.org	317-908-8737x05134	2020-08-25	53573
9	2	4	Juan	Banks	barbara46@example.org	976.390.3978x909	2022-06-16	63701
10	2	3	Virginia	Robinson	dean58@example.org	819-764-5975x6869	2022-02-10	53439
11	5	2	Robert	Ramos	wwolf@example.net	001-876-831-9176x3...	2022-09-12	61618
12	4	1	Holly	Zimmerman	ballalexander@example.net	543.635.1215	2020-07-04	75081
13	3	3	Elizabeth	Martin	rogerpatterson@example.com	(256)515-4572x89578	2024-03-11	59653

employees 1 × | jobs 2 | departments 3

```
5 •  SELECT * FROM dwh_tp.jobs;
```

Result Grid | Filter Rows: | Export:

job_id	job_name	min_salary	max_salary
1	Manager	50000	80000
2	Developer	40000	60000
3	Designer	35000	50000
4	Analyst	45000	70000
5	Assistant	30000	35000

```
7 •  SELECT * FROM dwh_tp.departments;
```

Result Grid | Filter Rows: | Expo

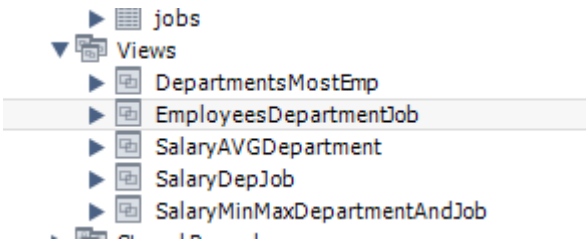
dep_id	dep_name
1	Human Resources
2	Finance
3	Marketing
4	Operations
5	Sales

DESAFIO #4:

Dependiendo si escoge una base de datos SQL/NoSQL, un Datawarehouse, o un Datalake, entonces desarrolle una view/query/report a partir del modelo de datos

Solución:

Se crean 5 vistas con el fin de generar una serie de reportes para ser invocadas desde el frontend de streamlit.




```sql
CREATE VIEW EmployeesDepartmentJob AS
SELECT e.employee_id, e.first_name,e.last_name, e.email,e.phone_number,e.contract_date, d.dep_name, j.job_name
FROM employees e
INNER JOIN departments d ON e.department_id = d.dep_id
INNER JOIN jobs j ON e.job_id = j.job_id;
```

```sql
#--------------------Vista de Salarios por Departamento y Puesto de Trabajo: va
CREATE VIEW SalaryMinMaxDepartmentAndJob AS
SELECT d.dep_name, j.job_name, MIN(e.salary) AS min_salary, MAX(e.salary) AS max_salary
FROM employees e
INNER JOIN departments d ON e.department_id = d.dep_id
INNER JOIN jobs j ON e.job_id = j.job_id
GROUP BY d.dep_name, j.job_name;
```

```sql
CREATE VIEW DepartmentsMostEmp AS
SELECT d.dep_name, COUNT(e.employee_id) AS num_employees
FROM employees e
INNER JOIN departments d ON e.department_id = d.dep_id
GROUP BY d.dep_name
ORDER BY num_employees DESC;
```

```sql
CREATE VIEW SalaryAVGDepartment AS
SELECT d.dep_name, COUNT(e.employee_id) AS num_employees, AVG(e.salary) AS avg_salary
FROM employees e
INNER JOIN departments d ON e.department_id = d.dep_id
GROUP BY d.dep_name;
```

```sql
create view SalaryDepJob as
SELECT d.dep_name, j.job_name,
       MIN(e.salary) AS min_salary,
       MAX(e.salary) AS max_salary,
       AVG(e.salary) AS avg_salary,
       COUNT(e.employee_id) AS num_employees
FROM employees e
INNER JOIN departments d ON e.department_id = d.dep_id
INNER JOIN jobs j ON e.job_id = j.job_id
GROUP BY d.dep_name, j.job_name;
```

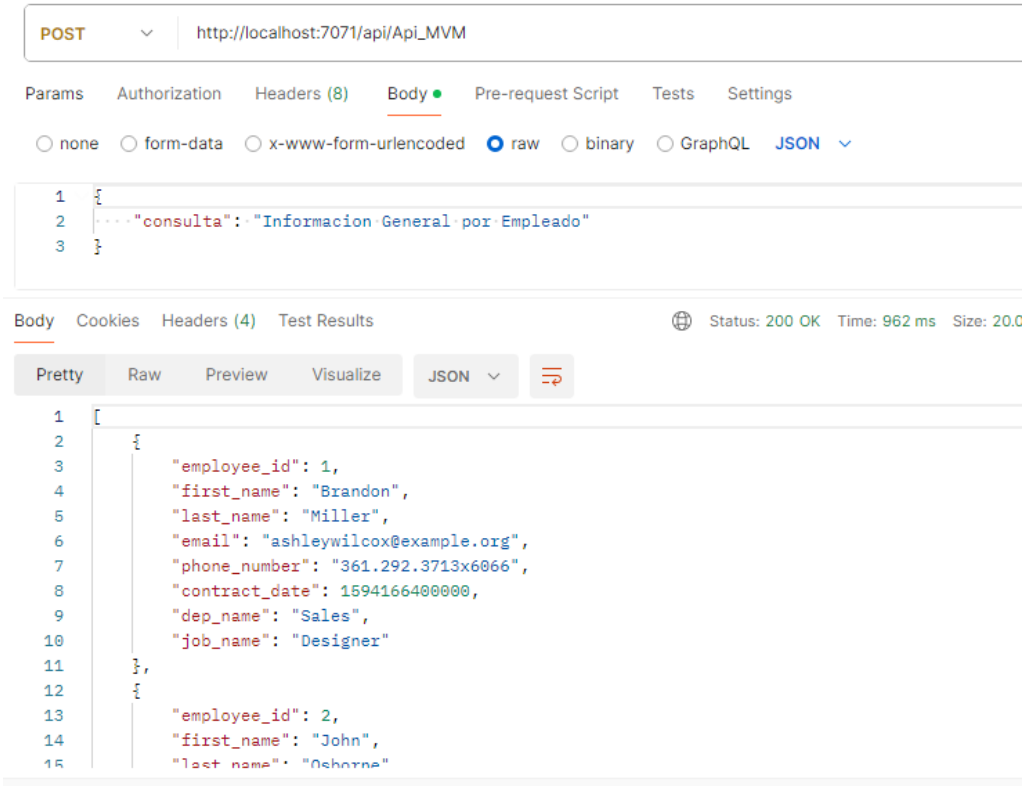
DESAFIO #5:

Desarrolle una API REST para consultar la view/query/report. Para el desarrollo de la API considere algún framework de Python, C#/.Net.

Solución :

Se crea un azure Function Http Trigger como orquestador de servicios, el cual es el encargado de orquestar las peticiones realizadas por el frontend desarrollado en streamlit, y gestionar el acceso a la base de datos para el consumo de las vistas desarrolladas anteriormente para la generación de reportes.

Se selecciona este servicio ya que es un servicio brindado por azure el cual es aprovisionado, escalable y mantenido de manera automática brindando un aprovisionamiento serverles evitando asi la administración de servidores o infraestructura.



DESAFIO #6:
Mejore la implementación de la API realizando un despliegue que use contenedores (valide las distintas opciones que le brinda su nube). Considere una prueba de consumo a la API implementando o activando algún front de acceso para ejecutar la invocación a la view/query/report.

Solución:
Se crea un frontend en streamlit el cual es el encargado de consumir el API desarrollada con un Azure Function Http Trigger.
Este frontend cuenta con un combo box el cual tiene definida una serie de opciones las cuales permite la generación de reportes teniendo en cuenta las vistas generadas en la base de datos.

Una vez se selecciona alguna de las opciónes, este se encarga de realizar una petición al api, teniendo en cuenta la vista correspondiente del reporte de ña base de datos mysql. Generando como resultado un reporte, el cual puede ser descargado como un archivo CSV por parte del usuario

	employee_id	first_name	last_name	email	phone_number	contract_
0	1	Brandon	Miller	ashleywilcox@example.org	361.292.3713x6066	1,594,16
1	2	John	Osborne	jimenezbrandon@example.com	752-398-5954x03932	1,600,21
2	3	Micheal	Brown	amypham@example.com	(342)281-4004x61433	1,611,18
3	4	Dean	Smith	hamiltontom@example.com	365-210-9127	1,684,62
4	5	Mark	Holloway	ycampbell@example.net	301-353-2266	1,712,79
5	6	Adam	Anthony	andersonlucas@example.org	(830)867-1970x8598	1,624,40
6	7	Amanda	Mcdonald	michaelmunoz@example.net	(216)746-8408x70241	1,691,02
7	8	Alan	Kennedy	william25@example.org	317-908-8737x05134	1,598,31
8	9	Juan	Banks	barbara46@example.org	976.390.3978x909	1,655,33
9	10	Virginia	Robinson	dean58@example.org	819-764-5975x6869	1,644,45

Consulta de Vistas MySQL

Selecciona una opción:

Salario por Departamento y Area Trabajo

Resultados:

	dep_name	job_name	min_salary	max_salary
0	Sales	Designer	54,413	71,460
1	Human Resources	Manager	56,586	78,059
2	Finance	Designer	53,439	79,661
3	Marketing	Analyst	34,255	69,673
4	Operations	Developer	32,913	75,348
5	Operations	Assistant	37,458	76,870
6	Finance	Assistant	63,136	78,719
7	Human Resources	Analyst	53,573	71,667
8	Finance	Analyst	38,283	79,673
9	Sales	Developer	44,292	61,618

Download CSV

Consulta de Vistas MySQL

Selecciona una opción:

Cantidad Empleados por Departamento

Resultados:

	dep_name	num_employees
0	Finance	21
1	Marketing	21
2	Operations	21
3	Sales	20
4	Human Resources	17

Download CSV

Consulta de Vistas MySQL

Selecciona una opción:

Salario Promedio por Departamento

Resultados:

	dep_name	num_employees	avg_salary
0	Sales	20	57,688.3
1	Human Resources	17	57,875.2941
2	Finance	21	57,896.7143
3	Marketing	21	56,173.5238
4	Operations	21	54,850.9524

Download CSV

Consulta de Vistas MySQL

Selecciona una opción:

Informacion General de los Salarios

Resultados:

	dep_name	job_name	min_salary	max_salary	avg_salary	num_employees
0	Sales	Designer	54,413	71,460	63,524.75	4
1	Human Resources	Manager	56,586	78,059	65,530.3333	6
2	Finance	Designer	53,439	79,661	65,751	4
3	Marketing	Analyst	34,255	69,673	50,541	4
4	Operations	Developer	32,913	75,348	52,453.4286	7
5	Operations	Assistant	37,458	76,870	53,936	4
6	Finance	Assistant	63,136	78,719	70,927.5	2
7	Human Resources	Analyst	53,573	71,667	61,365	4
8	Finance	Analyst	38,283	79,673	58,000.1	10
9	Sales	Developer	44,292	61,618	54,461.6667	3

Download CSV