Kevin Fortier
SFL Take Home Assignment

1. **What is a Data Lake? Explain its benefits, how it differs from a data warehouse, and how it might benefit a client.**

A Data lake is a data storage solution where data is stored from various data sources in its raw form. In comparison, a data warehouse requires stored data to fit a specific schema. While a data warehouse enables greater ease of querying and analyzing stored data, it requires that the data schema and the types of queries be determined in advance. On the other hand, the burden of access and analyzing data stored in a data lake falls on the individual consumers of the data.

A data lake is especially useful for data coming from diverse data sources and/or does not easily fit into a predefined schema. This type of schema-less or unstructured data is commonly used in modern data analysis methods such as machine learning. Examples of unstructured data are audio files, images, text files and emails. Additionally, a data lake may also store structured data.

An organizational data initiative involving a data warehouse has a heavier upfront cost, as the data schema will first need to be determined. Determining a data schema for a data warehouse involves first understanding the types of use cases, queries and analysis required of the data as well as understanding all data sources and the transformations required to fit that data into the schema of the data warehouse.

An organizational data initiative involving a data lake has less of an upfront cost and enables quicker adoption. Data can be dumped into the data lake without worrying about the schema or how the data must be transformed.

2. **Explain serverless architecture. What are its pros and cons?**

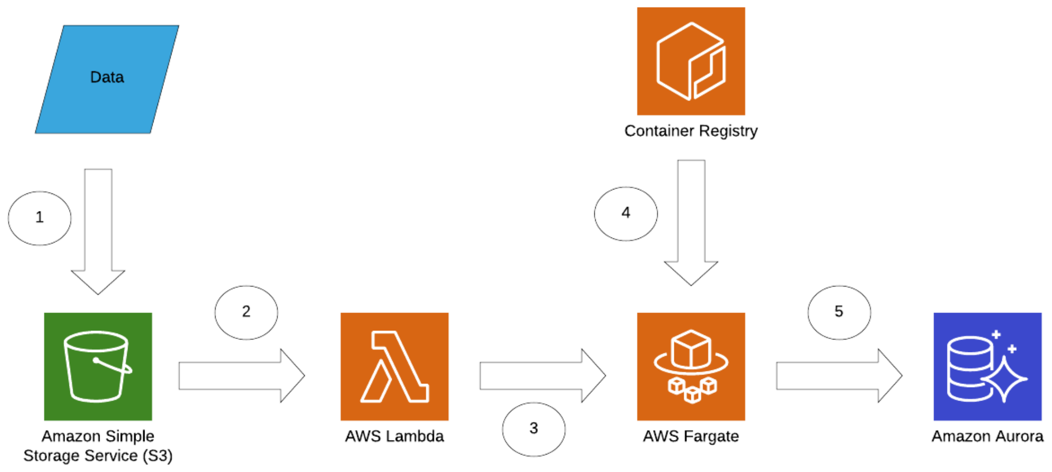
Serverless architecture is a pattern for developing software applications where infrastructure components are utilized from cloud providers. The serverless architectural pattern allows developers to focus on developing the core of the application without needing to manage the infrastructure or scalability of the application. As a result, serverless architecture can often lead to faster development time for developers and lower costs of running the application as serverless services often have a pay per usage pricing model.

With a serverless architecture, the final application will be tied to the cloud vendor on which the application was built. This is referred to as vendor lock in and can be seen as a negative because the vendor may make some change to the way in which their technology works or to their terms of service and the application owners can either choose to accept these changes or pay the heavy cost of rebuilding the system.

Another disadvantage of the serverless architecture pattern is multitenancy, where multiple applications, potentially from different users, are running on the same serverless infrastructure. Multitenancy could lead to exposure of sensitive data if the application is not configured properly.

To summarize, if development ease and velocity are of concern- serverless offers these advantages and should be considered as an architectural pattern. If avoiding vendor lockin and maintaining high security standards is a high priority then other architectures involving customized infrastructure should be considered

3. **Please provide a diagram of the ETL pipeline from Section 1 using serverless AWS services. Describe each component and its function within the pipeline.**



1. Data is uploaded to S3.
2. S3 invokes a Lambda function and passes in an S3 event.

3. The lambda function grabs the bucket and key that was uploaded from the S3 event, and invokes a task via AWS Fargate to run the etl pipeline. Running the etl pipeline on Fargate as opposed to lambda bypasses the shorter runtime limits of lambda which may be of concern for large data being uploaded.
4. The container for the pipeline task is pulled from the Container Registry, where containers are stored for access by other AWS services.
5. Faregate executes a task defined in a Docker container, which it pulls from the container registry (4). The container task contains the pipeline code. It will read the data from the file uploaded to S3, transform it in some way and load it into a database. The pipeline loads the transformed data into an Amazon Aurora database- a serverless scalable relational database.

Description of Components:
S3- Used as a data storage solution. It is used here as a landing spot for data that is uploaded for processing.
AWS Lambda- Function as a service that integrates with other AWS services and can respond to events from those services. Here, our lambda function is listening to events from S3 when new data is uploaded for processing.
AWS Fargate- Runs container based tasks and applications. The system above is running the etl pipeline within a container.
Container Registry- Service for storing containers.
Amazon Aurora- Scalable Relational Database.

4. Describe modern MLOps and how organizations should be approaching management from a tool and system perspective.

Building a successful machine learning solution is an iterative process and involves a full life cycle consisting of multiple stages where data is collected, the model is developed and trained, the model's performance is evaluated and the model architecture is adjusted and/or more data is collected to address specific weak areas of the model. After the model has reached sufficient performance levels it is integrated into a production software system and deployed.

The process of developing a machine learning model does not stop after the model has been deployed into production. It is best to monitor the model's performance in production and will likely require continued adjustment to the model itself and/or the data used to train the model. Thus, the entire lifecycle of building a machine learning system is one that will occur not once but many times. For this reason, it is important to follow a set of repeatable steps and processes in the management of this development lifecycle.

MLOPS refers to the set of best practices that an organization can follow to ensure success in building ML systems. Importantly, MLOPs is not dependent on any particular programming language, framework, platform or infrastructure. Therefore, organizations should seek to adopt an understanding of the key principles, stages and processes of the machine learning lifecycle, as described above, in order to guide their MLOPS implementations.

MLOPS seeks to leverage the understanding and best practices used in devops for shipping successful software products to shipping successful ML products. Some of the key components are automated testing, validation and deployment of critical software and artifacts produced in the ML development lifecycle. These practices create a consistent and repeatable strategy for developing, deploying and improving ML systems.

Again, MLOPS is not a specific set of technologies or any particular platform. Thus, it is important to assess how these practices can be implemented within the context of the tools and systems currently being used within your organization either for developing existing ML products and/or the software systems in which the desired ML solutions will be integrated.