

## QUESTION 1

A company is developing a le-sharing application that will use an Amazon S3 bucket for storage. The company wants to serve all the les through an Amazon CloudFront distribution. The company does not want the les to be accessible through direct navigation to the S3 URL.

What should a solutions architect do to meet these requirements?

- A. Write individual policies for each S3 bucket to grant read permission for only CloudFront access.
- B. Create an IAM user. Grant the user read permission to objects in the S3 bucket. Assign the user to CloudFront.
- C. Write an S3 bucket policy that assigns the CloudFront distribution ID as the Principal and assigns the target S3 bucket as the Amazon Resource Name (ARN).
- D. Create an origin access identity (OAI). Assign the OAI to the CloudFront distribution. Con gure the S3 bucket permissions so that only the OAI has read permission.

### Soru:

Bir şirket, depolama için **Amazon S3 bucket** kullanacak bir **dosya paylaşım uygulaması** geliştirmektedir. Şirket, tüm dosyaları **Amazon CloudFront distribution** üzerinden sunmak istemektedir. Şirket, dosyaların **doğrudan S3 URL'sine gidilerek erişilebilir olmasını istememektedir**.

Bir **solutions architect**, bu gereksinimleri karşılamak için ne yapmalıdır?

- A. Yalnızca CloudFront erişimine izin vermek için her bir S3 bucket için ayrı ayrı politikalar yazmalıdır.
- B. Bir **IAM kullanıcı** oluşturmalı, bu kullanıcıya S3 bucket içindeki nesnelere okuma izni vermelii ve kullanıcıyı CloudFront'a atamalıdır.
- C. **CloudFront distribution ID**'yi *Principal* olarak atanayan ve hedef S3 bucket'ı **Amazon Resource Name (ARN)** olarak tanımlayan bir **S3 bucket policy** yazmalıdır.
- D. Bir **Origin Access Identity (OAI)** oluşturmalı, bu OAI'yi CloudFront distribution'a atamalii ve S3 bucket izinlerini yalnızca OAI'nin okuma iznine sahip olacak şekilde yapılandırmalıdır.

### Soru Analizi:

Soruda istenenler çok net:

1. **Dosyalar S3'te tutulacak**
2. **Tüm erişim CloudFront üzerinden olacak**
3. **✗ S3 URL'si ile doğrudan erişim OLMAYACAK**

👉 Yani amaç:

**S3'ü private tut, sadece CloudFront erişebilsin**

Bu, AWS sınavlarında **klasik ve çok net bir senaryodur.**

🔑 **Anahtar Kelimeler (Exam Trigger)**

İfade	Anlam
“serve all files through CloudFront”	CDN zorunlu
“not accessible through direct S3 URL”	❗ S3 <b>private</b> olmalı
“CloudFront + S3”	<b>OAI / OAC</b>

⚠️ Sınavlarda bu ifade → %99 OAI (veya yeni nesil OAC)

**Sonuç Analizi:**

✅ **D. Create an Origin Access Identity (OAI)**

**Neden doğru?**

- **OAI**, CloudFront'un S3'e private erişimi için özel kimlidir
- Bucket policy:
  - ❌ Public access kapalı
  - ✅ Sadece OAI read permission
- Böylece:
  - ✅ CloudFront erişir
  - ❌ S3 URL ile erişim engellenir

📌 **AWS best practice (exam):**

**S3 + CloudFront + private content → OAI**

❌ **A. Write individual policies for each S3 bucket to grant read permission for only CloudFront access**

**Neden yanlış?**

- “CloudFront” tek başına bir IAM principal değildir
- CloudFront'a doğrudan “allow” veremezsin
- **OAI veya OAC olmadan CloudFront S3'ten private obje çekemez**

📌 Exam trick:

“Grant read permission to CloudFront” → **eksik / yanlış ifade**

✗ **B. Create an IAM user and assign it to CloudFront**

**Neden yanlış?**

- CloudFront **IAM user kullanmaz**
- IAM user → uygulamalar / insanlar içindir
- CDN ile IAM user eşleştirme diye bir kavram yok

📌 Exam trick:

“IAM user + CloudFront” → **direkt ele**

✗ **C. Use CloudFront distribution ID as Principal in S3 bucket policy**

**Neden yanlış?**

- CloudFront Distribution ID **Principal olamaz**
- Principal; IAM user, role, OAI veya OAC olabilir
- Teknik olarak **geçersiz yapılandırıma**

📌 Exam trick:

“Distribution ID as principal” → ✗

🎯 Sonuç

🧠 **Sınavda Hatırlaman İçin Altın Cümle**

“**S3 private, CloudFront public**”

→ **OAI (veya yeni sorularda OAC)**

📅 **Güncel Not (Bonus Bilgi)**

AWS artık **OAC (Origin Access Control)** öneriyor.

Ama:

- Sınav sorularının büyük kısmı hâlâ **OAI** bekler
- Şıklar arasında **OAC yoksa** → **OAI seç**

---

## QUESTION 2

A company's website provides users with downloadable historical performance reports. The website needs a solution that will scale to meet the company's website demands globally. The solution should be cost-effective, limit the provisioning of infrastructure resources, and provide the fastest possible response time.

Which combination should a solutions architect recommend to meet these requirements?

- A. Amazon CloudFront and Amazon S3
- B. AWS Lambda and Amazon DynamoDB
- C. Application Load Balancer with Amazon EC2 Auto Scaling
- D. Amazon Route 53 with internal Application Load Balancers

**Soru:**

Bir şirketin web sitesi, kullanıcılarla indirilebilir **geçmiş performans raporları** sunmaktadır. Web sitesinin, şirketin **küresel ölçekteki taleplerini karşılayacak şekilde ölçeklenebilir** bir çözüme ihtiyacı vardır. Çözüm **maliyet etkin** olmalı, **altyapı kaynaklarının önceden sağlanması (provisioning)** sınırlandırmalı ve **mükemmel olan en hızlı yanıt süresini** sağlamalıdır.

Bu gereksinimleri karşılamak için bir **çözümler mimarı** aşağıdaki hangi kombinasyonu önermelidir?

- A. **Amazon CloudFront ve Amazon S3**
- B. **AWS Lambda ve Amazon DynamoDB**
- C. **Application Load Balancer ile Amazon EC2 Auto Scaling**
- D. **Amazon Route 53 ile dahili (internal) Application Load Balancer'lar**

**Soru Analizi:**

**Temel Gereksinimler**

Soruda verilen **anahtar ifadeler**:

1. **Downloadable historical performance reports**
  - Statik içerik (dosyalar, raporlar)
  - Dinamik işlem yok
2. **Scale globally**
  - Küresel erişim
  - Düşük gecikme (low latency)
3. **Cost-effective**

- Ucuz çözüm
- Gereksiz kaynak çalıştırılmamak

#### 4. Limit provisioning of infrastructure resources

- Sunucu kurmamak / yönetmemek
- Serverless veya managed servisler

#### 5. Fastest possible response time

- CDN kullanımı
- Kullanıcıya en yakın noktadan içerik sunma

👉 Bu gereksinimler **statik içerik + küresel dağıtım + CDN + serverless** çözümü işaret eder.

#### Sonuç Analizi:

##### ✓ A. Amazon CloudFront ve Amazon S3

###### 🧠 En Uygun Mimari Ne Olmalı?

- **Statik dosyalar → Amazon S3**
- **Küresel ve hızlı dağıtım → Amazon CloudFront (CDN)**
- **Sunucu yok → düşük maliyet + otomatik ölçeklenme**

- ✓ Statik dosyalar S3'te tutulur
- ✓ CloudFront ile global edge location'lardan servis edilir
- ✓ Çok düşük gecikme
- ✓ Otomatik ölçeklenir
- ✓ Sunucu yönetimi yok
- ✓ En düşük maliyetli çözümlerden biri

###### ● Tüm gereksinimleri eksiksiz karşılar

##### ✗ B. AWS Lambda ve Amazon DynamoDB

- ✗ DynamoDB dosya saklamak için uygun değil
- ✗ Lambda, dosya dağıtımını için gereksiz karmaşık
- ✗ CDN yok → gecikme yüksek

##### ● Statik dosya indirme senaryosu için yanlış

##### ✗ C. Application Load Balancer + EC2 Auto Scaling

- ✖ EC2 instance'ları sürekli çalışır → maliyetli
- ✖ Altyapı provisioning gereklidir
- ✖ Global dağıtım için ek yapı gereklidir
- ✖ CDN yok → gecikme daha yüksek

### ● Gereksiz sunucu yönetimi

#### ✖ D. Amazon Route 53 + Internal ALB

- ✖ Internal ALB internete açık değildir
- ✖ Kullanıcılar erişemez
- ✖ Dosya dağıtımları için uygun değil

### ● Teknik olarak yanlış

#### 🎯 Sonuç

Eğer soruda şu kelimeleri görürsen:

- *downloadable*
- *historical*
- *static content*
- *global*
- *fast response*
- *cost-effective*

👉 Aklına ilk gelen çözüm: S3 + CloudFront olmalı.

---

## QUESTION 3

A company runs an Oracle database on premises. As part of the company's migration to AWS, the company wants to upgrade the database to the most recent available version. The company also wants to set up disaster recovery (DR) for the database. The company needs to minimize the operational overhead for normal operations and DR setup. The company also needs to maintain access to the database's underlying operating system.

Which solution will meet these requirements?

- Migrate the Oracle database to an Amazon EC2 instance. Set up database replication to a different AWS Region.
- Migrate the Oracle database to Amazon RDS for Oracle. Activate Cross-Region automated backups to replicate the snapshots to another AWS Region.

C. Migrate the Oracle database to Amazon RDS Custom for Oracle. Create a read replica for the database in another AWS Region.

D. Migrate the Oracle database to Amazon RDS for Oracle. Create a standby database in another Availability Zone.

**Soru:**

Bir şirket, şirket içinde (on-premises) **Oracle veritabanı** çalışmaktadır. Şirketin AWS'e geçişi kapsamında, veritabanını **mevcut en güncel sürümü yükseltmek** istemektedir. Şirket ayrıca veritabanı için **felaket kurtarma (Disaster Recovery – DR)** kurmak istemektedir. Şirket, **normal operasyonlar ve DR kurulumunda operasyonel yükü en aza indirmek** istemektedir. Ayrıca şirketin, veritabanının **altındaki işletim sistemine (operating system) erişimini sürdürmesi** gerekmektedir.

Bu gereksinimleri karşılayacak çözüm hangisidir?

A. Oracle veritabanını bir **Amazon EC2** instance'ına taşıyın. Veritabanı çoğaltmasını (replication) farklı bir **AWS Region**'a kurun.

B. Oracle veritabanını **Amazon RDS for Oracle**'a taşıyın. Snapshot'ları başka bir **AWS Region**'a kopyalamak için **Cross-Region automated backups** özelliğini etkinleştirin.

C. Oracle veritabanını **Amazon RDS Custom for Oracle**'a taşıyın. Veritabanı için başka bir **AWS Region**'da bir **read replica** oluşturun.

D. Oracle veritabanını **Amazon RDS for Oracle**'a taşıyın. Başka bir **Availability Zone**'da bir **standby database** oluşturun.

**Soru Analizi:**

**Sorudaki Kritik Gereksinimler (Anahtar Kelimeler)**

**1. Oracle database (on-premises)**

- Oracle lisans ve sürüm yönetimi önemli

**2. Upgrade to the most recent available version**

- Veritabanı sürüm yükseltme desteği
- AWS tarafından yönetilen upgrade tercih edilir

**3. Set up disaster recovery (DR)**

- Region veya AZ seviyesinde yedeklilik
- Otomatik ve yönetilen bir DR çözümü

**4. Minimize operational overhead**

- OS patching, backup, replication gibi işlemleri AWS yönetsin

## 5. Maintain access to the underlying operating system

- **⚠ En kritik gereksinim**
- Standart **Amazon RDS for Oracle** bunu sağlamaz
- **RDS Custom for Oracle** sağlar

👉 Bu gereksinimler birlikte düşünüldüğünde:

- **EC2** → OS erişimi var ama operasyonel yük yüksek
- **RDS for Oracle** → operasyonel yük düşük ama OS erişimi yok
- **RDS Custom for Oracle** → düşük operasyonel yük + OS erişimi ✓

### Sonuç Analizi:

#### ✓ C. Oracle → Amazon RDS Custom for Oracle + Cross-Region Read Replica

- ✓ OS seviyesinde erişim mevcut
- ✓ AWS tarafından yönetilen altyapı
- ✓ En güncel Oracle sürümleri desteklenir
- ✓ Read Replica ile **Region-level DR**
- ✓ Operasyonel yük EC2'ye göre çok düşük

#### ● Tüm gereksinimleri karşılayan tek seçenek

#### ✗ A. Oracle → Amazon EC2 + Cross-Region Replication

- ✓ OS erişimi var
- ✗ Backup, patching, replication, DR tamamen manuel
- ✗ Yüksek operasyonel yük
- ✗ Upgrade süreci karmaşık

#### ● “Minimize operational overhead” şartını karşılamaz

#### ✗ B. Oracle → Amazon RDS for Oracle + Cross-Region Backups

- ✓ Operasyonel yük düşük
- ✓ Snapshot tabanlı DR mümkün
- ✗ **OS erişimi yok**
- ✗ Standby / aktif DR değil, sadece backup

#### ● En kritik gereksinimi kaçırıyor (OS access)

## D. Oracle → Amazon RDS for Oracle + Multi-AZ Standby

- ✓ Otomatik failover
- ✓ Operasyonel yük düşük
-  **OS erişimi yok**
-  Multi-AZ ≠ Region-level DR (sadece AZ failure)

### DR ve OS erişimi açısından yetersiz

### Sonuç

Gereksinim	Doğru Servis
OS erişimi	EC2 veya <b>RDS Custom</b>
Düşük operasyonel yük	<b>RDS / RDS Custom</b>
Oracle + OS erişimi	<b>RDS Custom for Oracle</b>
Region-level DR	Cross-Region Read Replica

---

## QUESTION 4

A company wants to move its application to a serverless solution. The serverless solution needs to analyze existing and new data by using SL. The company stores the data in an Amazon S3 bucket. The data requires encryption and must be replicated to a different AWS Region.

Which solution will meet these requirements with the LEAST operational overhead?

- A. Create a new S3 bucket. Load the data into the new S3 bucket. Use S3 Cross-Region Replication (CRR) to replicate encrypted objects to an S3 bucket in another Region. Use server-side encryption with AWS KMS multi-Region keys (SSE-KMS). Use Amazon Athena to query the data.
- B. Create a new S3 bucket. Load the data into the new S3 bucket. Use S3 Cross-Region Replication (CRR) to replicate encrypted objects to an S3 bucket in another Region. Use server-side encryption with AWS KMS multi-Region keys (SSE-KMS). Use Amazon RDS to query the data.
- C. Load the data into the existing S3 bucket. Use S3 Cross-Region Replication (CRR) to replicate encrypted objects to an S3 bucket in another Region. Use server-side encryption with Amazon S3 managed encryption keys (SSE-S3). Use Amazon Athena to query the data.

D. Load the data into the existing S3 bucket. Use S3 Cross-Region Replication (CRR) to replicate encrypted objects to an S3 bucket in another Region. Use server-side encryption with Amazon S3 managed encryption keys (SSE-S3). Use Amazon RDS to query the data.

**Soru:**

Bir şirket, uygulamasını **sunucusuz (serverless)** bir çözüme taşımak istemektedir. Bu sunucusuz çözümün, **mevcut ve yeni verileri makine öğrenmesi (ML)** kullanarak analiz etmesi gerekmektedir. Şirket, verileri bir **Amazon S3 bucket** içinde saklamaktadır. Verilerin **şifrelenmesi ve farklı bir AWS Region'a çoğaltılması (replication)** gerekmektedir. Şirket, **en az operasyonel yük (least operational overhead)** ile bu gereksinimleri karşılayan çözümü istemektedir.

Bu gereksinimleri karşılayan çözüm hangisidir?

A. Yeni bir **S3 bucket** oluşturun. Verileri bu yeni S3 bucket'a yükleyin. Şifrelenmiş nesneleri başka bir Region'daki S3 bucket'a çoğaltmak için **S3 Cross-Region Replication (CRR)** kullanın. **AWS KMS çok bölgeli anahtarları (multi-Region keys)** ile **server-side encryption (SSE-KMS)** kullanın. Verileri sorgulamak için **Amazon Athena** kullanın.

B. Yeni bir **S3 bucket** oluşturun. Verileri bu yeni S3 bucket'a yükleyin. Şifrelenmiş nesneleri başka bir Region'daki S3 bucket'a çoğaltmak için **S3 Cross-Region Replication (CRR)** kullanın. **AWS KMS çok bölgeli anahtarları (multi-Region keys)** ile **server-side encryption (SSE-KMS)** kullanın. Verileri sorgulamak için **Amazon RDS** kullanın.

C. Verileri **mevcut S3 bucket** içine yükleyin. Şifrelenmiş nesneleri başka bir Region'daki S3 bucket'a çoğaltmak için **S3 Cross-Region Replication (CRR)** kullanın. **Amazon S3 tarafından yönetilen şifreleme anahtarları (SSE-S3)** ile **server-side encryption** kullanın. Verileri sorgulamak için **Amazon Athena** kullanın.

D. Verileri **mevcut S3 bucket** içine yükleyin. Şifrelenmiş nesneleri başka bir Region'daki S3 bucket'a çoğaltmak için **S3 Cross-Region Replication (CRR)** kullanın. **Amazon S3 tarafından yönetilen şifreleme anahtarları (SSE-S3)** ile **server-side encryption** kullanın. Verileri sorgulamak için **Amazon RDS** kullanın.

**Soru Analizi:**

**Anahtar Gereksinimler**

1. **Serverless solution**

- Sunucu yönetimi olmamalı
- EC2 / RDS gibi servisler tercih edilmemeli

## **2. Analyze existing and new data**

- Veriler S3'te duruyor
- SQL benzeri sorgular / analiz gerekiyor

## **3. Data is stored in Amazon S3**

- Ekstra veri taşıma minimum olmalı
- Mevcut bucket kullanımı avantaj

## **4. Data requires encryption**

- AWS tarafından yönetilen şifreleme yeterli
- “En az operasyonel yük” → manuel anahtar yönetimi olmamalı

## **5. Replicated to a different AWS Region**

- **S3 Cross-Region Replication (CRR)** gereklidir

## **6. LEAST operational overhead (en kritik ifade)**

- Tam yönetilen (fully managed)
- Sunucusuz
- Minimum konfigürasyon

### **Sonuç Analizi:**

 **C. Mevcut S3 bucket + CRR + SSE-S3 + Athena**

### **İdeal Mimari Ne Olmalıdır?**

- **Depolama** → Amazon S3
- **Şifreleme** → SSE-S3 (anahtar yönetimi yok)
- **Replication** → S3 CRR
- **Analiz / sorgulama** → Amazon Athena (serverless)

### **✓ Tamamen serverless**

- ✓ Mevcut altyapı kullanılır
- ✓ SSE-S3 → anahtar yönetimi yok
- ✓ CRR → Region-level replication
- ✓ Athena → serverless sorgulama

 **Tüm gereksinimleri en düşük operasyonel yükle karşılar**

 **A. Yeni S3 bucket + CRR + SSE-KMS (multi-Region) + Athena**

- ✓ Serverless
- ✓ Athena doğru
- ✗ KMS multi-Region keys → ekstra yönetim
- ✗ Yeni bucket → gereksiz ek operasyon
- ✗ “Least operational overhead” için fazla karmaşık

🟡 Çalışır ama optimal değil

✗ B. Yeni S3 bucket + CRR + SSE-KMS + Amazon RDS

✗ Amazon RDS serverless değildir

✗ Veriyi RDS'e yükleme ve yönetme gerekir

✗ Yüksek operasyonel yük

🔴 Serverless şartını net olarak karşılamaz

✗ D. Mevcut S3 bucket + CRR + SSE-S3 + Amazon RDS

✗ Amazon RDS serverless değil

✗ Database yönetimi gerekir

✗ En yüksek operasyonel yük

🔴 Yanlış

🎯 Sonuç

#### Gereksinim

Serverless analytics **Amazon Athena**

S3 verisi                  **Yerinde analiz (no ETL)**

En az operasyon      **SSE-S3**

Region replication    **S3 CRR**

Soruda “least operational overhead” görürsen:

👉 KMS yerine SSE-S3,

👉 RDS yerine Athena düşün.

---

#### QUESTION 5

A company runs workloads on AWS. The company needs to connect to a service from an external provider. The service is hosted in the provider's VPC. According to the

company's security team, the connectivity must be private and must be restricted to the target service. The connection must be initiated only from the company's VPC.

Which solution will meet these requirements?

- A. Create a VPC peering connection between the company's VPC and the provider's VPC. Update the route table to connect to the target service.
- B. Ask the provider to create a virtual private gateway in its VPC. Use AWS PrivateLink to connect to the target service.
- C. Create a NAT gateway in a public subnet of the company's VPC. Update the route table to connect to the target service.
- D. Ask the provider to create a VPC endpoint for the target service. Use AWS PrivateLink to connect to the target service.

#### Soru:

Bir şirket, AWS üzerinde iş yükleri (workloads) çalıştırmaktadır. Şirket, **harici bir sağlayıcıya ait bir servise** bağlanması gerekmektedir. Bu servis, sağlayıcının **kendi VPC'si** içinde barındırılmaktadır. Şirketin güvenlik ekibine göre, bağlantının **özel (private)** olması ve **yalnızca hedef servise erişimle sınırlanması** gerekmektedir. Ayrıca bağlantı **yalnızca şirketin VPC'si tarafından başlatılabilmesi** gerekmektedir.

Bu gereksinimleri karşılayan çözüm hangisidir?

- A. Şirketin VPC'si ile sağlayıcının VPC'si arasında bir **VPC peering bağlantısı** oluşturun. Hedef servise bağlanmak için **route table'ları** güncelleyin.
- B. Sağlayıcıdan, kendi VPC'sinde bir **virtual private gateway** oluşturmasını isteyin. Hedef servise bağlanmak için **AWS PrivateLink** kullanın.
- C. Şirketin VPC'sindeki bir **public subnet** içinde bir **NAT gateway** oluşturun. Hedef servise bağlanmak için **route table'ları** güncelleyin.
- D. Sağlayıcıdan, hedef servis için bir **VPC endpoint** oluşturmasını isteyin. Hedef servise bağlanmak için **AWS PrivateLink** kullanın.

#### Soru Analizi:

##### Kritik Gereksinimler (Anahtar İfadeler)

1. **External provider's service**
  - Servis başka bir firmaya ait
  - Servis, sağlayıcının **kendi VPC'sinde**
2. **Connectivity must be private**

- Internet üzerinden erişim yasak
- Public IP / NAT / IGW olmamalı

### 3. Restricted to the target service

- Sağlayıcının VPC'sindeki **diğer kaynaklara erişim olmamalı**
- Sadece belirli bir servise erişim

### 4. Connection must be initiated only from the company's VPC

- Tek yönlü erişim
- Provider VPC → Company VPC erişimi olmamalı

👉 Bu gereksinimler **AWS PrivateLink**'i net şekilde işaret eder.

#### 🧠 Doğru Mimari Ne Olmalı?

- **Sağlayıcı tarafı** → Servisi bir **VPC Endpoint Service** olarak yayınlar
- **Şirket tarafı** → **Interface VPC Endpoint** oluşturur
- Trafik:
  - Özel AWS ağı üzerinden
  - Tek yönlü (consumer → provider)
  - Sadece o servise

#### Sonuç Analizi:

##### ✓ D. Provider VPC endpoint + AWS PrivateLink

- ✓ Tamamen private
- ✓ Sadece hedef servise erişim
- ✓ Tek yönlü bağlantı
- ✓ En yüksek güvenlik
- ✓ AWS önerilen çözüm

##### ● Tüm gereksinimleri karşılayan tek seçenek

##### ✗ A. VPC Peering

- ✓ Private bağlantı
- ✗ Tüm VPC'ler birbirini görür
- ✗ Route table ile geniş erişim

Servisle sınırlı değil

Karşı VPC erişimi de mümkün

**Güvenlik gereksinimlerini karşılamaz**

**B. Virtual Private Gateway + PrivateLink**

Virtual Private Gateway → VPN / Direct Connect için

PrivateLink ile birlikte kullanılmaz

Kavramsal olarak yanlış

**Teknik olarak hatalı**

**C. NAT Gateway**

Internet çıkışı sağlar

Private bağlantı değildir

Servis kısıtlaması yok

**Tamamen yanlış**

**Sonuç**

**Gereksinim      Çözüm**

3rd-party service    **AWS PrivateLink**

Private + restricted **Interface VPC Endpoint**

One-way access    **PrivateLink**

No VPC visibility    **PrivateLink (no peering)**

Soruda şu ifadeleri görürsen:

- *external provider*
- *restricted to a service*
- *private connection*
- *initiated only from company VPC*

**Cevap %90 ihtimalle: AWS PrivateLink**

---

**QUESTION 6**

A company is migrating its on-premises PostgreSQL database to Amazon Aurora PostgreSQL. The on-premises database must remain online and accessible during the migration. The Aurora database must remain synchronized with the on-premises database.

Which combination of actions must a solutions architect take to meet these requirements? (Choose two.)

- A. Create an ongoing replication task.
- B. Create a database backup of the on-premises database.
- C. Create an AWS Database Migration Service (AWS DMS) replication server.
- D. Convert the database schema by using the AWS Schema Conversion Tool (AWS SCT).
- E. Create an Amazon EventBridge (Amazon CloudWatch Events) rule to monitor the database synchronization.

**Soru:**

Bir şirket, şirket içi (on-premises) **PostgreSQL** veritabanını **Amazon Aurora PostgreSQL**'e taşıyor. Şirket içi veritabanı, taşıma sırasında **çevrimiçi (online) kalmalı ve erişilebilir olmalıdır**. Aurora veritabanı, şirket içi veritabanı ile **senkronize** kalmalıdır.

Bir **çözümler mimarı**, bu gereksinimleri karşılamak için aşağıdaki işlemlerden hangilerinin kombinasyonunu gerçekleştirmelidir? (**İki seçenek seçin.**)

- A. Sürekli (ongoing) bir replikasyon görevi oluşturmak
- B. Şirket içi veritabanının bir yedekini oluşturmak
- C. Bir **AWS Database Migration Service (AWS DMS)** replikasyon sunucusu oluşturmak
- D. **AWS Schema Conversion Tool (AWS SCT)** kullanarak veritabanı şemasını dönüştürmek
- E. Veritabanı senkronizasyonunu izlemek için bir **Amazon EventBridge (Amazon CloudWatch Events)** kuralı oluşturmak

**Soru Analizi:**

**Temel gereksinimler:**

1. **On-premises PostgreSQL veritabanı online kalmalı**
  - Yani *downtime* olmadan taşıma yapılmalı.
2. **Amazon Aurora PostgreSQL ile sürekli senkronizasyon olmalı**
  - Yani *one-time migration* değil, **continuous replication** gerekiyor.
3. Hedef veritabanı: **Amazon Aurora PostgreSQL**

#### 4. Kaynak veritabanı: **On-premises PostgreSQL**

Bu gereksinimler AWS'de klasik olarak **AWS Database Migration Service (DMS)** ile karşılanır.

##### **Doğru Yaklaşım (Yüksek Seviye)**

- **AWS DMS** kullanılır
- **Ongoing (continuous) replication** açılır
- Böylece:
  - Kaynak DB online kalır
  - Değişiklikler (CDC – Change Data Capture) Aurora'ya anlık aktarılır

##### **Sonuç Analizi:**

###### **A. Create an ongoing replication task**

###### **✓ DOĞRU**

- AWS DMS'te **ongoing replication**, CDC (Change Data Capture) anlamına gelir.
- Kaynak veritabanı çalışmaya devam ederken:
  - INSERT
  - UPDATE
  - DELETEİşlemleri hedef veritabanına aktarılır.
- **Senkronizasyon şartını doğrudan karşılar.**

 Bu seçenek **zorunludur**.

###### **C. Create an AWS Database Migration Service (AWS DMS) replication server**

###### **✓ DOĞRU**

- AWS DMS çalışabilmek için mutlaka:
  - **Replication instance (server)** ister
- Bu instance:
  - Kaynak ve hedef DB arasında veri taşır
- Ongoing replication task bu sunucu üzerinde çalışır.

 DMS kullanıyorsan bu **olmazsa olmazdır**.

## B. Create a database backup of the on-premises database

### X YANLIŞ

- Backup:
  - One-time snapshot'tır
  - Senkronizasyon sağlamaz
- Ayrıca:
  - Büyük veritabanlarında downtime gerekebilir
- Soruda özellikle:

*must remain online and synchronized*  
deniyor.

 Backup, bu senaryo için **yetersizdir**.

## D. Convert the database schema by using the AWS Schema Conversion Tool (AWS SCT)

### X YANLIŞ (Bu senaryo için)

- AWS SCT:
  - **Heterojen** migration'larda gereklidir
  - Örn: Oracle → PostgreSQL
- Burada:
  - PostgreSQL → Aurora PostgreSQL
  - Yani **homojen migration**
- Şema büyük ölçüde uyumludur.

 Zorunlu değil, bu yüzden yanlış.

## E. Create an Amazon EventBridge rule to monitor the database synchronization

### X YANLIŞ

- Monitoring:
  - Operasyonel bir iyileştirmedir
  - Gereksinim değildir
- Senkronizasyonu **sağlamaz**, sadece izler.

 Sorunun ana hedeflerine hizmet etmiyor.

 **Sonuç**

- ✓ **A. Create an ongoing replication task**
- ✓ **C. Create an AWS DMS replication server**

 **Kısa Özeti (Sınav İpucu)**

**Online + Senkron + On-prem → AWS = AWS DMS + Ongoing Replication**

---

## QUESTION 7

A company uses AWS Organizations to create dedicated AWS accounts for each business unit to manage each business unit's account independently upon request. The root email recipient missed a notification that was sent to the root user email address of one account. The company wants to ensure that all future notifications are not missed. Future notifications must be limited to account administrators.

Which solution will meet these requirements?

- A. Configure the company's email server to forward notification email messages that are sent to the AWS account root user email address to all users in the organization.
- B. Configure all AWS account root user email addresses as distribution lists that go to a few administrators who can respond to alerts. Configure AWS account alternate contacts in the AWS Organizations console or programmatically.
- C. Configure all AWS account root user email messages to be sent to one administrator who is responsible for monitoring alerts and forwarding those alerts to the appropriate groups.
- D. Configure all existing AWS accounts and all newly created accounts to use the same root user email address. Configure AWS account alternate contacts in the AWS Organizations console or programmatically.

### Soru:

Bir şirket, her iş biriminin hesabını talep üzerine bağımsız olarak yönetebilmesi için **AWS Organizations** kullanarak her iş birimi için **ayrı AWS hesapları** oluşturuyor. Bir hesabın **root kullanıcı e-posta adresine** gönderilen bir bildirim, root e-posta alıcısı fark etmedi (kaçırdı). Şirket, gelecekte gönderilecek hiçbir bildirimin kaçırılmamasını istiyor. Gelecekteki bildirimler **yalnızca hesap yöneticileriyle (account administrators)** sınırlı olmalıdır.

Bu gereksinimleri karşılayan çözüm hangisidir?

- A. AWS hesap root kullanıcı e-posta adresine gönderilen bildirim e-postalarını, organizasyondaki tüm kullanıcılar iletecek şekilde şirketin e-posta sunucusunu yapılandırmak
- B. Tüm AWS hesaplarının root kullanıcı e-posta adreslerini, uyarılara yanıt verebilecek birkaç yöneticiyi içeren **dağıtım listeleri** olarak yapılandırmak. AWS Organizations konsolundan veya programatik olarak **AWS hesap alternatif iletişim bilgilerini (alternate contacts)** yapılandırmak
- C. Tüm AWS hesap root kullanıcı e-posta mesajlarını, uyarıları izlemekten sorumlu tek bir yöneticiye göndermek ve bu yöneticinin uyarıları uygun grplara iletmesini sağlamak
- D. Mevcut tüm AWS hesaplarının ve yeni oluşturulan tüm hesapların **aynı root kullanıcı e-posta adresini** kullanmasını sağlamak. AWS Organizations konsolundan veya programatik olarak **AWS hesap alternatif iletişim bilgilerini (alternate contacts)** yapılandırmak

#### Soru Analizi:

##### Temel problemler:

- **Root user e-postası kritik bildirimler alır**
  - Fatura, güvenlik, hesap kapatma, servis kısıtlamaları vb.
- Bildirim **kaçırılmış**
- Amaç:
  1. **Gelecekte hiçbir bildirimin kaçırılmaması**
  2. Bildirimler **sadece hesap yöneticilerine (account administrators)** gitmeli
  3. **Merkezi ama kontrollü** bir yapı
  4. AWS Organizations kullanılıyor (çoklu hesap ortamı)

#### Sonuç Analizi:

- B. Root e-postalarını dağıtım listesi yapmak + Alternate Contacts yapılandırmak**

#### AWS'in Önerdiği En İyi Uygulama (Best Practice)

- Root user e-postası:
  - **Dağıtım listesi (distribution list)** olmalı
- Ayrıca:
  - **Alternate Contacts (Billing, Security, Operations)** tanımlanmalı
- Bu yapı:

- Tek kişiye bağımlılığı ortadan kaldırır
- Bildirimlerin ilgili **yöneticilere** gitmesini sağlar
- AWS Organizations ile merkezi yönetimi destekler

### Neden doğru?

- Root e-posta adresi:
  - Birden fazla **yetkili yöneticiye** gider
- Alternate Contacts:
  - Billing
  - Security
  - Operations
- AWS Organizations üzerinden:
  - Merkezi
  - Ölçeklenebilir
  - Standartlaştırılmış yapı

👉 Tüm gereksinimleri eksiksiz karşılar.

### ✗ A. Root e-postalarını organizasyondaki tüm kullanıcılarla iletmek

#### X YANLIŞ

- Gereksinim:

*Notifications must be limited to account administrators*

  - Bu seçenek:
    - Bildirimleri **tüm kullanıcılarla** gönderir
    - Güvenlik ve yetki prensiplerine aykırı

👉 Gereksinimi ihlal eder.

### ✗ C. Tüm root e-postalarını tek bir yöneticiye göndermek

#### X YANLIŞ

- Tek noktadan hata (Single Point of Failure)
- Bu kişi:
  - İzinli olabilir

- E-postayı kaçırabilir
- AWS best practice:
  - **Dağıtım listesi**, tek kişi değil

👉 Risklidir.

## ✗ D. Tüm hesapların aynı root e-posta adresini kullanması

### X YANLIŞ

- AWS her hesap için:
  - **Benzersiz root e-posta** ister
- Ayrıca:
  - Hesap ayrimı ve güvenlik zayıflar
- AWS Organizations mantığına ters

👉 Pratikte ve teoride yanlış.

### 🎯 Sonuç

## AWS Organizations + Root email kaçırılmasın

### 👉 Distribution List + Alternate Contacts

## Root Email için En İyi Uygulama (Best Practice)

- **Root user email = Distribution List**
  - Örnek: aws-root-prod@company.com
- Listeye sadece:
  - Cloud Admin
  - Security Lead
  - Billing Owner

✗ Tek kişi

✗ Tüm kullanıcılar

✓ Birden fazla yetkili yönetici

## Sınavda Nasıl Gelir? (Altın Kural)

## Root email kaçırılmasın + AWS Organizations

- Distribution List
  - Alternate Contacts
  - Tek kişi
  - Tüm kullanıcılar
- 

## QUESTION 8

A company runs its ecommerce application on AWS. Every new order is published as a message in a RabbitMQ queue that runs on an Amazon EC2 instance in a single Availability Zone. These messages are processed by a different application that runs on a separate EC2 instance. This application stores the details in a PostgreSQL database on another EC2 instance. All the EC2 instances are in the same Availability Zone. The company needs to redesign its architecture to provide the highest availability with the least operational overhead.

What should a solutions architect do to meet these requirements?

- A. Migrate the queue to a redundant pair (active/standby) of RabbitMQ instances on Amazon MQ. Create a Multi-AZ Auto Scaling group for EC2 instances that host the application. Create another Multi-AZ Auto Scaling group for EC2 instances that host the PostgreSQL database.
- B. Migrate the queue to a redundant pair (active/standby) of RabbitMQ instances on Amazon MQ. Create a Multi-AZ Auto Scaling group for EC2 instances that host the application. Migrate the database to run on a Multi-AZ deployment of Amazon RDS for PostgreSQL.
- C. Create a Multi-AZ Auto Scaling group for EC2 instances that host the RabbitMQ queue. Create another Multi-AZ Auto Scaling group for EC2 instances that host the application. Migrate the database to run on a Multi-AZ deployment of Amazon RDS for PostgreSQL.
- D. Create a Multi-AZ Auto Scaling group for EC2 instances that host the RabbitMQ queue. Create another Multi-AZ Auto Scaling group for EC2 instances that host the application. Create a third Multi-AZ Auto Scaling group for EC2 instances that host the PostgreSQL database

### Soru:

Bir şirket, e-ticaret uygulamasını AWS üzerinde çalışmaktadır. Her yeni sipariş, **tek bir Availability Zone'da** çalışan bir **Amazon EC2** örneği üzerindeki **RabbitMQ kuyruğuna** bir mesaj olarak gönderilmektedir. Bu mesajlar, **ayrı bir EC2 üzerinde** çalışan farklı bir uygulama tarafından işlenmektedir. Bu uygulama, sipariş detaylarını **başka bir EC2 üzerinde** çalışan **PostgreSQL** veritabanına kaydetmektedir. Tüm EC2 örnekleri **aynı**

**Availability Zone** içindedir. Şirket, **en yüksek kullanılabilirliği (highest availability)** ve **en az operasyonel yükü (least operational overhead)** sağlayacak şekilde mimarisini yeniden tasarlamak istemektedir.

Bir çözümler mimarı bu gereksinimleri karşılamak için ne yapmalıdır?

A. Kuyruğu, **Amazon MQ üzerinde çalışan yedekli (aktif/standby) bir RabbitMQ çiftine** taşımak. Uygulamayı barındıran EC2 örnekleri için **Multi-AZ Auto Scaling grubu** oluşturmak. PostgreSQL veritabanını barındıran EC2 örnekleri için başka bir **Multi-AZ Auto Scaling grubu** oluşturmak.

B. Kuyruğu, **Amazon MQ üzerinde çalışan yedekli (aktif/standby) bir RabbitMQ çiftine** taşımak. Uygulamayı barındıran EC2 örnekleri için **Multi-AZ Auto Scaling grubu** oluşturmak. Veritabanını **Amazon RDS for PostgreSQL Multi-AZ dağıtımına** taşımak.

C. RabbitMQ kuyruğunu barındıran EC2 örnekleri için **Multi-AZ Auto Scaling grubu** oluşturmak. Uygulamayı barındıran EC2 örnekleri için başka bir **Multi-AZ Auto Scaling grubu** oluşturmak. Veritabanını **Amazon RDS for PostgreSQL Multi-AZ dağıtımına** taşımak.

D. RabbitMQ kuyruğunu barındıran EC2 örnekleri için **Multi-AZ Auto Scaling grubu** oluşturmak. Uygulamayı barındıran EC2 örnekleri için başka bir **Multi-AZ Auto Scaling grubu** oluşturmak. PostgreSQL veritabanını barındıran EC2 örnekleri için üçüncü bir **Multi-AZ Auto Scaling grubu** oluşturmak.

#### Soru Analizi:

##### Mevcut mimarideki problemler

- **Single Availability Zone**
  - AZ arızasında **tam kesinti**
- **RabbitMQ EC2 üzerinde**
  - Kendi HA, patch, failover yönetimi gerekiyor
- **PostgreSQL EC2 üzerinde**
  - Yedeklilik, backup, failover tamamen şirkete ait
- **Operasyonel yük yüksek**

##### Gereksinimler

1. **Highest availability**
  - AZ arızalarına dayanıklı olmalı
2. **Least operational overhead**

- Mümkürn olduğunda **managed services**
- 3. Mesajlaşma + uygulama + veritabanı
- 4. Multi-AZ mimari

#### Sonuç Analizi:

B

#### AWS Best Practice'e Göre Doğru Yaklaşım

**Katman**      **En iyi çözüm**

Message Queue **Amazon MQ (managed RabbitMQ, active/standby)**

Application      **EC2 Auto Scaling Group (Multi-AZ)**

Database      **Amazon RDS for PostgreSQL (Multi-AZ)**

Bu yaklaşım:

- AZ failure'a dayanıklı
- Failover otomatik
- Patch / backup / replication AWS tarafından yönetilir

Amazon MQ + App ASG + RDS PostgreSQL Multi-AZ

#### Neden doğru?

- **Amazon MQ**
  - Managed RabbitMQ
  - Active/standby, Multi-AZ
- **Application ASG**
  - Multi-AZ, self-healing
- **RDS PostgreSQL Multi-AZ**
  - Otomatik failover
  - Backup, patch, replication AWS yönetir

En yüksek availability + en az operasyonel yük

A

Amazon MQ + App ASG + PostgreSQL EC2 ASG

### Neden yanlış?

- Veritabanı hâlâ **EC2 üzerinde**
- DB failover, replication, backup:
  - Operasyonel yük yüksek
- RDS varken EC2 DB **best practice değil**

 *Least operational overhead* karşılanmasıyor.

 **C**

RabbitMQ EC2 ASG + App ASG + RDS

### Neden yanlış?

- RabbitMQ hâlâ EC2 üzerinde
- RabbitMQ cluster yönetimi:
  - Zor
  - Operasyonel yük yüksek
- Amazon MQ varken EC2 RabbitMQ **tercih edilmez**

 **D**

Her şey EC2 ASG

### Neden yanlış?

- Tamamen self-managed
- En yüksek operasyonel yük
- AWS managed servis avantajı yok

 **Sonuç**

 **Sınavda Altın Kural**

**Highest availability + least operational overhead**

- ✓ Managed services
- ✓ Multi-AZ
-  EC2 üzerinde DB
-  EC2 üzerinde MQ (managed alternatifleri varken)

---

## QUESTION 9

A reporting team receives les each day in an Amazon S3 bucket. The reporting team manually reviews and copies the les from this initial S3 bucket to an analysis S3 bucket each day at the same time to use with Amazon QuickSight. Additional teams are starting to send more les in larger sizes to the initial S3 bucket. The reporting team wants to move the les automatically analysis S3 bucket as the les enter the initial S3 bucket.

The reporting team also wants to use AWS Lambda functions to run pattern-matching code on the copied data. In addition, the reporting team wants to send the data les to a pipeline in Amazon SageMaker Pipelines.

What should a solutions architect do to meet these requirements with the LEAST operational overhead?

- A. Create a Lambda function to copy the les to the analysis S3 bucket. Create an S3 event notification for the analysis S3 bucket. Configure Lambda and SageMaker Pipelines as destinations of the event notification. Configure s3:ObjectCreated:Put as the event type.
- B. Create a Lambda function to copy the les to the analysis S3 bucket. Configure the analysis S3 bucket to send event notifications to Amazon EventBridge (Amazon CloudWatch Events). Configure an ObjectCreated rule in EventBridge (CloudWatch Events). Configure Lambda and SageMaker Pipelines as targets for the rule.
- C. Configure S3 replication between the S3 buckets. Create an S3 event notification for the analysis S3 bucket. Configure Lambda and SageMaker Pipelines as destinations of the event notification. Configure s3:ObjectCreated:Put as the event type.
- D. Configure S3 replication between the S3 buckets. Configure the analysis S3 bucket to send event notifications to Amazon EventBridge (Amazon CloudWatch Events). Configure an ObjectCreated rule in EventBridge (CloudWatch Events). Configure Lambda and SageMaker Pipelines as targets for the rule.

#### Soru:

Bir raporlama ekibi, her gün **Amazon S3** üzerindeki bir bucket'a dosyalar almaktadır. Raporlama ekibi, bu **ilk (initial) S3 bucket**'ndaki dosyaları her gün aynı saatte **manuel olarak inceleyip analiz S3 bucket**'ına kopyalamaktadır ve bu dosyaları **Amazon QuickSight** ile kullanmaktadır. Ek ekipler, ilk S3 bucket'ına **daha fazla sayıda ve daha büyük boyutlu dosyalar** göndermeye başlamıştır. Raporlama ekibi, dosyalar ilk S3 bucket'ına girdikçe **otomatik olarak analiz S3 bucket**'ına taşınmasını istemektedir.

Raporlama ekibi ayrıca, kopyalanan veriler üzerinde **AWS Lambda fonksiyonları** kullanarak **pattern-matching (desen eşleştirme)** kodu çalışırmak istemektedir. Buna ek olarak, raporlama ekibi **veri dosyalarını Amazon SageMaker Pipelines içindeki bir pipeline'a göndermek** istemektedir.

Bu gereksinimleri **EN AZ operasyonel yük (least operational overhead)** ile karşılamak için bir çözümler mimarı ne yapmalıdır?

- A. Dosyaları analiz S3 bucket'ına kopyalamak için bir **Lambda fonksiyonu** oluşturmak. Analiz S3 bucket'ı için bir **S3 event notification** oluşturmak. **Lambda ve SageMaker Pipelines**'ı bu event notification'ın hedefleri olarak yapılandırmak. Event tipi olarak **s3:ObjectCreated:Put** yapılandırmak.
- B. Dosyaları analiz S3 bucket'ına kopyalamak için bir **Lambda fonksiyonu** oluşturmak. Analiz S3 bucket'ını **Amazon EventBridge (Amazon CloudWatch Events)**'e event gönderecek şekilde yapılandırmak. EventBridge (CloudWatch Events) üzerinde bir **ObjectCreated** kuralı oluşturmak. **Lambda ve SageMaker Pipelines**'ı bu kuralın hedefleri olarak yapılandırmak.
- C. İki S3 bucket arasında **S3 replication** yapılandırmak. Analiz S3 bucket'ı için bir **S3 event notification** oluşturmak. **Lambda ve SageMaker Pipelines**'ı bu event notification'ın hedefleri olarak yapılandırmak. Event tipi olarak **s3:ObjectCreated:Put** yapılandırmak.
- D. İki S3 bucket arasında **S3 replication** yapılandırmak. Analiz S3 bucket'ını **Amazon EventBridge (Amazon CloudWatch Events)**'e event gönderecek şekilde yapılandırmak. EventBridge (CloudWatch Events) üzerinde bir **ObjectCreated** kuralı oluşturmak. **Lambda ve SageMaker Pipelines**'ı bu kuralın hedefleri olarak yapılandırmak.

#### Soru Analizi:

#### Mevcut durum

- Dosyalar **initial (ilk) S3 bucket**'ına geliyor
- Raporlama ekibi:
  - Dosyaları **manuel** olarak
  - **analysis S3 bucket**'ına kopyalıyor
- Dosyalar:
  - **Artan sayıda**
  - **Daha büyük boyutlarda**

#### Gereksinimler

1. **Dosyalar otomatik olarak kopyalansın**
  - Dosya S3'e girer girmez
2. **Büyük dosyalar desteklensin**

3. Lambda ile pattern-matching çalıştırılsın
4. Veriler Amazon SageMaker Pipelines'a gönderilsin
5. En az operasyonel yük (least operational overhead)
  - Managed servisler
  - Minimum kod
  - Minimum bakım

**Sonuç Analizi:**

 **D. S3 Replication + EventBridge**

**En Uygun Mimari Yaklaşım**

- **S3 Replication**
  - Dosya kopyalama için yerleşik ve tamamen yönetilen çözüm
- **Amazon EventBridge**
  - Tek bir olaydan birden fazla hedefe (Lambda + SageMaker Pipelines) yönlendirme
- **S3 → EventBridge → Targets**
  - Ölçeklenebilir
  - Esnek
  - Düşük operasyonel yük

**Neden doğru?**

- **S3 Replication**
  - Büyük dosyalar için ideal
  - Tamamen managed
- **Amazon EventBridge**
  - Lambda ve SageMaker Pipelines aynı anda target olabilir

**En az operasyonel yük sağlanır**

 **A. Lambda ile kopyalama + S3 Event Notification**

**Neden yanlış?**

- Büyük dosyaların kopyalanması Lambda ile:

- Timeout riski
- Maliyet artışı
- S3 Event Notification:
  - Çoklu hedeflerde sınırlı
- Operasyonel yük **yüksek**

## B. Lambda ile kopyalama + EventBridge

### Neden yanlış?

- EventBridge doğru bir servis
- Ancak:
  - Kopyalama hâlâ Lambda ile yapılıyor
- S3 Replication varken:
  - Lambda ile kopyalama **best practice değil**
- Operasyonel yük **gereksiz**

## C. S3 Replication + S3 Event Notification

### Neden yanlış?

- S3 Replication doğru
- Ancak:
  - S3 Event Notification hedef seçenekleri sınırlı
  - SageMaker Pipelines entegrasyonu zayıf
- EventBridge varken **daha az esnek**

## Sonuç

## Sınav İpucu

### S3 dosya kopyalama + büyük dosyalar + least ops

- ✓ S3 Replication
- ✓ EventBridge
- ✗ Lambda ile kopyalama

---

## QUESTION 10

A solutions architect needs to help a company optimize the cost of running an application on AWS. The application will use Amazon EC2 instances, AWS Fargate, and AWS Lambda for compute within the architecture.

The EC2 instances will run the data ingestion layer of the application. EC2 usage will be sporadic and unpredictable. Workloads that run on EC2 instances can be interrupted at any time. The application front end will run on Fargate, and Lambda will serve the API layer. The front-end utilization and API layer utilization will be predictable over the course of the next year.

Which combination of purchasing options will provide the MOST cost-effective solution for hosting this application? (Choose two.)

- A. Use Spot Instances for the data ingestion layer
- B. Use On-Demand Instances for the data ingestion layer
- C. Purchase a 1-year Compute Savings Plan for the front end and API layer.
- D. Purchase 1-year All Upfront Reserved instances for the data ingestion layer.
- E. Purchase a 1-year EC2 instance Savings Plan for the front end and API layer.

**Soru:**

Bir **Solutions Architect**, bir şirketin AWS üzerinde çalışan bir uygulamanın maliyetini optimize etmesine yardımcı olmalıdır. Uygulama mimarisinde **hesaplama (compute)** için **Amazon EC2, AWS Fargate ve AWS Lambda** kullanacaktır.

**EC2 instance'ları**, uygulamanın **veri alma (data ingestion) katmanını** çalıştıracaktır. EC2 kullanımı **düzensiz ve tahmin edilemezdir**. EC2 instance'ları üzerinde çalışan iş yükleri **herhangi bir zamanda kesintiye uğrayabilir**. Uygulamanın **ön yüzü (front end)** **Fargate** üzerinde çalışacaktır ve **API katmanı Lambda** tarafından sağlanacaktır. Ön yüzün ve API katmanının kullanımı, **önümüzdeki bir yıl boyunca öngörülebilirdir**.

Bu uygulamayı barındırmak için **EN maliyet etkin çözümü** sağlayacak **satin alma seçenekleri kombinasyonu** hangisidir? (**İki tanesini seçin.**)

- A. Veri alma katmanı için **Spot Instance** kullanmak
- B. Veri alma katmanı için **On-Demand Instance** kullanmak
- C. Ön yüz ve API katmanı için **1 yıllık Compute Savings Plan** satın almak
- D. Veri alma katmanı için **1 yıllık All Upfront Reserved Instance** satın almak
- E. Ön yüz ve API katmanı için **1 yıllık EC2 Instance Savings Plan** satın almak

**Soru Analizi:**

**Uygulama 3 farklı compute servisi kullanıyor:**

**EC2 – Data Ingestion Layer**

- **Kullanım:** Düzensiz, öngörülemez
- **Kesinti toleransı:** Var (iş yükleri her an durdurulabilir)  
→ En önemli ipuçları:

*sporadic, unpredictable, can be interrupted*

📌 Bu ifadeler AWS sınavlarında Spot Instance’ı işaret eder.

#### Fargate – Front End

- **Kullanım:** Öngörülebilir
- **Süre:** En az 1 yıl
- **Servis türü:** EC2 değil, container tabanlı

#### Lambda – API Layer

- **Kullanım:** Öngörülebilir
- **Servis türü:** Serverless, EC2 bağımsız

📌 Fargate + Lambda birlikte düşünülmeli ve EC2’ye özel çözümlerden kaçınılmalıdır.

#### 🧠 Doğru Satın Alma Stratejisi Mantığı

- Kesintiye dayanıklı + öngörülemez → Spot Instances
- Öngörülebilir kullanım + EC2 dışı servisler → Compute Savings Plan

#### Seçenek Analizi:

✓ A. Use Spot Instances for the data ingestion layer

✓ Doğru

- EC2 kullanımı düzensiz
- İş yükleri kesintiye dayanıklı
- Spot, %70–90’'a varan maliyet avantajı sağlar  
→ En maliyet etkin seçenek

✗ C. Purchase a 1-year Compute Savings Plan for the front end and API layer

✓ Doğru

- Compute Savings Plan:
  - Fargate
  - Lambda

- **EC2**

- Öngörülebilir kullanım için idealdır
- En esnek ve maliyet avantajı yüksek plan

➔ AWS sınavlarında **Lambda + Fargate** varsa → **Compute Savings Plan**

✗ **B. Use On-Demand Instances for the data ingestion layer**

✗ **Yanlış**

- On-Demand pahalıdır
- Öngörülemez ama **kesintiye dayanıklı** iş yükleri için Spot varken tercih edilmez

✗ **D. Purchase 1-year All Upfront Reserved instances for the data ingestion layer**

✗ **Yanlış**

- Reserved Instances:
  - Uzun süreli
  - Sürekli çalışan
  - Kesintisiz iş yükleri içindir
- Sorudaki EC2 kullanımı **düzensiz ve kesintiye açık**

✗ **E. Purchase a 1-year EC2 Instance Savings Plan for the front end and API layer**

✗ **Yanlış**

- EC2 Instance Savings Plan **sadece EC2** içindir
- **Fargate ve Lambda'yı kapsamaz**

🎯 **Sonuç**

- *Can be interrupted* → **Spot**
- *Predictable usage* → **Savings Plan**
- *Lambda / Fargate* → **Compute Savings Plan**
- *Unpredictable + RI* → ✗

Senaryo / İpucu (Soruda Geçen İfade)	En Uygun Seçenek Neden	
<i>Can be interrupted</i>	<b>Spot Instance</b>	En ucuz, kesinti tolere edilir
<i>Sporadic / unpredictable</i>	<b>Spot Instance</b>	Süre taahhüdü yok
<i>Short-term / test / dev</i>	<b>On-Demand</b>	Bağlayıcılık yok
<i>Steady / predictable / long-term</i>	<b>Reserved Instance</b>	Uzun vadede ucuz
<i>Always on, 24/7</i>	<b>Reserved Instance</b>	Sürekli kullanım
<i>Cost optimization (general)</i>	<b>Savings Plan</b>	RI'den daha esnek



### SINAV ALTIN KURALLARI

- **Interruptible** görürsen → **Spot**
- **Predictable (1 yıl / 3 yıl)** görürsen → **Savings Plan**
- **Lambda / Fargate** görürsen → **Compute Savings Plan**
- **Unpredictable + RI** → **X**
- **EC2 Instance SP + Lambda** → **X**

### QUESTION 11

A company runs a web-based portal that provides users with global breaking news, local alerts, and weather updates. The portal delivers each user a personalized view by using mixture of static and dynamic content. Content is served over HTTPS through an API server running on an Amazon EC2 instance behind an Application Load Balancer (ALB). The company wants the portal to provide this content to its users across the world as quickly as possible.

How should a solutions architect design the application to ensure the LEAST amount of latency for all users?

- Deploy the application stack in a single AWS Region. Use Amazon CloudFront to serve all static and dynamic content by specifying the ALB as an origin.
- Deploy the application stack in two AWS Regions. Use an Amazon Route 53 latency routing policy to serve all content from the ALB in the closest Region.

C. Deploy the application stack in a single AWS Region. Use Amazon CloudFront to serve the static content. Serve the dynamic content directly from the ALB.

D. Deploy the application stack in two AWS Regions. Use an Amazon Route 53 geolocation routing policy to serve all content from the ALB in the closest Region.

#### Soru:

Bir şirket, kullanıcılarla **küresel son dakika haberleri, yerel uyarılar ve hava durumu güncellemeleri** sağlayan **web tabanlı bir portal** işletmektedir. Portal, **statik ve dinamik içeriğin bir karışımını** kullanarak her kullanıcıya **kişiselleştirilmiş bir görünüm** sunmaktadır.

İçerik, **Application Load Balancer (ALB)** arkasında çalışan bir **Amazon EC2 instance** üzerindeki **API sunucusu** aracılığıyla **HTTPS** üzerinden sunulmaktadır. Şirket, portalın bu içeriği **dünya genelindeki kullanıcılarla mümkün olan en hızlı şekilde** sunmasını istemektedir.

Bir **Solutions Architect**, tüm kullanıcılar için **EN AZ gecikmeyi (latency)** sağlamak amacıyla uygulamayı nasıl tasarlamalıdır?

A. Uygulama yiğinini tek bir AWS Region'da dağıtmak. Tüm statik ve dinamik içeriği sunmak için **ALB'yi origin olarak belirleyerek Amazon CloudFront** kullanmak.

B. Uygulama yiğinini iki AWS Region'da dağıtmak. Tüm içeriği en yakın Region'daki ALB'den sunmak için **Amazon Route 53 latency routing policy** kullanmak.

C. Uygulama yiğinini tek bir AWS Region'da dağıtmak. **Statik içeriği Amazon CloudFront** ile sunmak. **Dinamik içeriği doğrudan ALB üzerinden** sunmak.

D. Uygulama yiğinini iki AWS Region'da dağıtmak. Tüm içeriği en yakın Region'daki ALB'den sunmak için **Amazon Route 53 geolocation routing policy** kullanmak.

#### Soru Analizi:

##### 📌 Uygulamanın Temel Özellikleri

- **Global kullanıcı kitlesi** 
- **Statik + dinamik içerik** birlikte sunuluyor
- İçerik **HTTPS** üzerinden
- **ALB arkasında EC2** üzerinde çalışan API
- Hedef: **Tüm kullanıcılar için EN DÜŞÜK latency**

##### 📌 Kritik Sınav İpuçları

Sorudaki anahtar ifadeler:

- *across the world*

- *as quickly as possible*
- *mixture of static and dynamic content*

### Sonuç Analizi:

#### ✓ A. Single Region + CloudFront (ALB origin)

👉 Bu üçlü AWS sınavlarında **CloudFront**'ı işaret eder.

✖ **CloudFront**, içeriği **edge location**'lardan sunarak hem statik **hem de dinamik** içeriğin gecikmesini azaltır.

#### 🧠 En Doğru Mimari Yaklaşım

- **Tek Region + CloudFront**
- **ALB → CloudFront origin**
- Statik içerik cache'lenir
- Dinamik içerik **edge üzerinden** ALB'ye en kısa yoldan iletilir

Bu, global latency'yi minimize eden **en etkili ve yaygın AWS çözümü**dür.

#### ✓ DOĞRU CEVAP

- CloudFront:
  - Statik içeriği cache'ler
  - Dinamik içeriği edge location'lardan ileter
- Tek Region bile olsa:
  - Global kullanıcılar en yakın edge'den hizmet alır
- En düşük latency + basit mimari

✖ **CloudFront sadece statik değil, dinamik içerik için de kullanılır** (sınav tuzağı).

#### ✗ B. Two Regions + Route 53 Latency Routing

#### ✗ Yanlış

- Route 53 latency routing:
  - DNS seviyesinde yönlendirme yapar
  - Edge cache yoktur
- Kullanıcı hâlâ ALB'ye kadar gider
- CloudFront'a göre **daha yüksek latency**

## C. CloudFront sadece statik, dinamik doğrudan ALB

### Yanlış

- Dinamik içerik CloudFront'tan faydalananmaz
- Global kullanıcılar ALB'ye uzun mesafe gider
- **Latency tam olarak minimize edilmez**

### Sınavda:

“Statik CloudFront, dinamik ALB” → eksik çözüm

## D. Two Regions + Route 53 Geolocation Routing

### Yanlış

- Geolocation routing:
  - Coğrafi kurallara göre yönlendirir
  - Latency'ye göre optimize etmez
- Edge cache yok
- CloudFront'tan daha yavaştır

### Sonuç

- **Global users + low latency** → CloudFront
- **Static + Dynamic content** → CloudFront (ikisi de)
- **Route 53** → DNS seviyesi (cache yok)
- **CloudFront origin** → ALB / EC2 / S3 olabilir

### Tek Cümlelik Ezber

Global + en düşük latency = CloudFront (ALB origin)

#### ◆ **Route 53 Routing Policy Karşılaştırması**

##### **Routing Policy Ne Yapar**

Latency      En düşük RTT'ye yönlendirir Cache sağlamaz

Geolocation      Ülke/kıta bazlı yönlendirir      Latency'ye bakmaz

Weighted      Trafik oranları      Global hızlandırma yapmaz

##### **Ne Yapmaz**

## QUESTION 12

A gaming company is designing a highly available architecture. The application runs on a modified Linux kernel and supports only UDP-based traffic. The company needs the front-end tier to provide the best possible user experience. That tier must have low latency, route traffic to the nearest edge location, and provide static IP addresses for entry into the application endpoints.

What should a solutions architect do to meet these requirements?

- A. Configure Amazon Route 53 to forward requests to an Application Load Balancer. Use AWS Lambda for the application in AWS Application Auto Scaling.
- B. Configure Amazon CloudFront to forward requests to a Network Load Balancer. Use AWS Lambda for the application in an AWS Application Auto Scaling group.
- C. Configure AWS Global Accelerator to forward requests to a Network Load Balancer. Use Amazon EC2 instances for the application in an EC2 Auto Scaling group.
- D. Configure Amazon API Gateway to forward requests to an Application Load Balancer. Use Amazon EC2 instances for the application in an EC2 Auto Scaling group.

**Soru:**

Bir oyun şirketi, yüksek erişilebilirliğe (**highly available**) sahip bir mimari tasarlamaktadır. Uygulama, **değiştirilmiş (modified)** bir **Linux çekirdeği** üzerinde çalışmaktadır ve **yalnızca UDP tabanlı trafiği** desteklemektedir. Şirket, **ön uç (front-end)** katmanının mümkün olan **en iyi kullanıcı deneyimini** sağlamasını istemektedir. Bu katman; **düşük gecikme (low latency)** sağlamalı, trafiği **en yakın edge location'a** yönlendirmeli ve uygulama uç noktalarına giriş için **statik IP adresleri** sağlamalıdır.

Bir **Solutions Architect**, bu gereksinimleri karşılamak için ne yapmalıdır?

- A. **Amazon Route 53** yapılandırarak istekleri **Application Load Balancer'a** yönlendirmek. Uygulama için **AWS Application Auto Scaling** içinde **AWS Lambda** kullanmak.
- B. **Amazon CloudFront** yapılandırarak istekleri **Network Load Balancer'a** yönlendirmek. Uygulama için **AWS Application Auto Scaling** içinde **AWS Lambda** kullanmak.
- C. **AWS Global Accelerator** yapılandırarak istekleri **Network Load Balancer'a** yönlendirmek. Uygulama için **EC2 Auto Scaling grubu** içinde **Amazon EC2 instance'ları** kullanmak.
- D. **Amazon API Gateway** yapılandırarak istekleri **Application Load Balancer'a** yönlendirmek. Uygulama için **EC2 Auto Scaling grubu** içinde **Amazon EC2 instance'ları** kullanmak.

**Soru Analizi:**

Sorudaki kritik gereksinimler:

1. **Highly available mimari**
2. **Sadece UDP trafiği**
3. **Modified Linux kernel**
  - o  **Lambda / managed servisler ELENİR**
4. **Low latency**
5. **Nearest edge location**
6. **Static IP addresses**

 Bu 3 ifade birlikte çok kritik:

*low latency + nearest edge location + static IP*

**Sonuç Analizi:**

 **C. Global Accelerator + NLB + EC2 ASG**

 AWS sınavlarında bu **doğrudan AWS Global Accelerator’ı** işaret eder.

 **Doğru Mimari Mantığı**

- **AWS Global Accelerator**
  - o Anycast **statik IP adresleri** sağlar
  - o Trafiği AWS global edge network’ü üzerinden taşıır
  - o Kullanıcıyı **en yakın edge location’a** yönlendirir
  - o **UDP ve TCP destekler**
- **Network Load Balancer**
  - o **UDP destekler**
  - o Düşük gecikme
  - o EC2 tabanlı uygulamalarla uyumlu
- **EC2 Auto Scaling**
  - o Modified kernel desteği
  - o Yüksek erişilebilirlik

 **DOĞRU CEVAP**

- Global Accelerator:

- Static IP ✓
  - Edge routing ✓
  - Low latency ✓
  - UDP support ✓
- NLB:
    - UDP support ✓
  - EC2:
    - Modified kernel ✓
  - Auto Scaling:
    - High availability ✓

### ✗ A. Route 53 + ALB + Lambda

- ALB ✗ UDP desteklemez
- Lambda ✗ modified kernel desteklemez
- Static IP ✗
  - Tamamen yanlış

### ✗ B. CloudFront + NLB + Lambda

- CloudFront ✗ UDP desteklemez
- Lambda ✗ modified kernel
- Static IP ✗
  - Yanlış

### ✗ D. API Gateway + ALB + EC2

- API Gateway ✗ UDP desteklemez
- ALB ✗ UDP desteklemez
- Static IP ✗
  - Yanlış

### ⌚ Sonuç

- **UDP + static IP + edge routing → AWS Global Accelerator**

- UDP load balancing → NLB
- Modified OS / kernel → EC2
- CloudFront / API Gateway / ALB → UDP desteklemez

### Tek Satırlık Ezber

*UDP + static IP + lowest latency = Global Accelerator + NLB*

- ◆ Global Accelerator vs CloudFront (Çok Karıştırılır)

Özellik	Global Accelerator	CloudFront
Trafik tipi	TCP ve UDP	Sadece HTTP/HTTPS
Static IP		
Gaming / real-time		
Edge routing		
Cache		
ALB/NLB backend	NLB / ALB	ALB / S3

Trafik tipi	TCP ve UDP	Sadece HTTP/HTTPS
Static IP		
Gaming / real-time		
Edge routing		
Cache		
ALB/NLB backend	NLB / ALB	ALB / S3

- ◆ Load Balancer Karşılaştırması

Load Balancer	UDP	HTTP/HTTPS	Static IP
ALB			
NLB			
Classic LB	sınırlı		

ALB			
NLB			
Classic LB	sınırlı		

### SINAV ALTIN KURALLARI

- UDP + static IP + edge → Global Accelerator
- UDP load balancing → NLB
- Gaming / real-time → Global Accelerator
- Lambda ≠ modified OS
- ALB / API Gateway ≠ UDP

### QUESTION 13

A company wants to migrate its existing on-premises monolithic application to AWS. The company wants to keep as much of the front-end code and the backend code as possible. However, the company wants to break the application into smaller applications. A different team will manage each application. The company needs a highly scalable solution that minimizes operational overhead.

Which solution will meet these requirements?

- A. Host the application on AWS Lambda. Integrate the application with Amazon API Gateway.
- B. Host the application with AWS Amplify. Connect the application to an Amazon API Gateway API that is integrated with AWS Lambda.
- C. Host the application on Amazon EC2 instances. Set up an Application Load Balancer with EC2 instances in an Auto Scaling group as targets.
- D. Host the application on Amazon Elastic Container Service (Amazon ECS). Set up an Application Load Balancer with Amazon ECS as the target.

**Soru:**

Bir şirket, mevcut **on-premises monolitik uygulamasını** AWS'ye taşımak istemektedir. Şirket, **ön uç (front-end) kodunun ve arka uç (back-end) kodunun** mümkün olduğunda büyük bir kısmını **korumak** istemektedir. Ancak şirket, uygulamayı **daha küçük uygulamalara** ayırmak istemektedir. Her bir uygulama **farklı bir ekip** tarafından yönetilecektir. Şirket, **operasyonel yükü en aza indiren, yüksek derecede ölçeklenebilir** bir çözüme ihtiyaç duymaktadır.

Bu gereksinimleri karşılayan çözüm hangisidir?

- A. Uygulamayı **AWS Lambda** üzerinde barındırmak. Uygulamayı **Amazon API Gateway** ile entegre etmek.
- B. Uygulamayı **AWS Amplify** ile barındırmak. Uygulamayı, **AWS Lambda ile entegre edilmiş bir Amazon API Gateway API'sine** bağlamak.
- C. Uygulamayı **Amazon EC2 instance'ları** üzerinde barındırmak. Hedef olarak **Auto Scaling grubundaki EC2 instance'larını** kullanan bir **Application Load Balancer** yapılandırmak.
- D. Uygulamayı **Amazon Elastic Container Service (Amazon ECS)** üzerinde barındırmak. Hedef olarak **Amazon ECS**'yi kullanan bir **Application Load Balancer** yapılandırmak.

**Soru Analizi:**

Sorudaki **kritik gereksinimler**:

1. **Mevcut monolitik uygulama**

- Front-end ve back-end kodunun **çoğu korunmak isteniyor**
2. **Uygulamayı daha küçük parçalara ayırma isteği**
    - → **Microservices yaklaşımı**
  3. **Her uygulama farklı ekip tarafından yönetilecek**
  4. **Yüksek ölçeklenebilirlik**
  5. **Operasyonel yükü en aza indirme**

📌 AWS sınavlarında bu kombinasyon şunu işaret eder:

### **Container tabanlı microservices (ECS / EKS)**

#### **Sonuç Analizi:**

##### **D. Amazon ECS + ALB**

##### **Doğru Mimari Mantığı**

- **Lambda:**
  - Mevcut kodu büyük ölçüde **yeniden yazmayı** gerektirir
- **EC2:**
  - Operasyonel yük **yüksek** (OS, patch, scaling)
- **Containers (ECS):**
  - Mevcut kod **minimum değişiklikle** taşınır
  - Her servis ayrı container
  - Ölçeklenebilir
  - ECS, EKS'e göre **daha az operasyonel yük**

##### **DOĞRU CEVAP**

- Container'lar:
  - Mevcut kod **küçük değişikliklerle** taşınır
  - Her microservice ayrı container
- ECS:
  - Managed service
  - Ölçeklenebilir
  - Düşük operasyonel yük

- ALB:
  - Servis bazlı routing

 **AWS sınav mantığı:**

*Monolith → microservices, minimal code change, low ops → ECS*

 **A. AWS Lambda + API Gateway**

- Monolit → Lambda geçisi **ciddi refactor** gerektirir
- Long-running / stateful uygulamalar için uygun değil
- Kodun “çoğunu koruma” şartına uymaz

 **B. AWS Amplify + API Gateway + Lambda**

- Amplify daha çok **frontend + serverless** içindir
- Backend yine Lambda → refactor gereklidir
- Monolit migration için uygun değil

 **C. EC2 + ALB + Auto Scaling**

- Lift-and-shift mümkün
- Ama:
  - Monolit kalır
  - Operasyonel yük yüksek
  - “Break into smaller applications” hedefini tam karşılamaz

 **Sonuç**

- **Keep existing code** → Containers
- **Break monolith** → Microservices
- **Low operational overhead** → ECS (EKS değil)
- **Lambda** → Refactor gereklidir

Kriter	ECS	EKS	Kriter
Operasyonel yük			Operasyonel yük
Exam preference			Exam preference
Cluster yönetimi	AWS yönetir	Kısmen müşteri	Cluster yönetimi

Kriter	ECS	EKS	Kriter
Yeni başlayanlar	✓	✗	Yeni başlayanlar

---

#### QUESTION 14

A company recently started using Amazon Aurora as the data store for its global ecommerce application. When large reports are run, developers report that the ecommerce application is performing poorly. After reviewing metrics in Amazon CloudWatch, a solutions architect finds that the ReadIOPS and CPUUtilization metrics are spiking when monthly reports run.

What is the MOST cost-effective solution?

- A. Migrate the monthly reporting to Amazon Redshift.
- B. Migrate the monthly reporting to an Aurora Replica.
- C. Migrate the Aurora database to a larger instance class.
- D. Increase the Provisioned IOPS on the Aurora instance.

#### Soru:

Bir şirket, **küresel e-ticaret uygulaması** için **Amazon Aurora**'yı veri deposu olarak kullanmaya yeni başlamıştır. **Büyük raporlar** çalıştırıldığında, geliştiriciler e-ticaret uygulamasının **kötü performans gösterdiğini** bildirmektedir. **Amazon CloudWatch** metrikleri incelediğinde, bir **Solutions Architect**, **aylık raporlar çalıştırıldığında ReadIOPS ve CPUUtilization** metriklerinin **ani artışlar (spike)** gösterdiğini tespit etmektedir.

Bu durum için **EN maliyet etkin çözüm** hangisidir?

- A. Aylık raporlamayı **Amazon Redshift**'e taşımak.
- B. Aylık raporlamayı bir **Aurora Replica**'ya taşımak.
- C. Aurora veritabanını **daha büyük bir instance sınıfına** taşımak.
- D. Aurora instance'ı için **Provisioned IOPS** değerini artırmak.

#### Soru Analizi:

##### Mevcut Durum

- **Amazon Aurora** → OLTP (işlemsel) veritabanı
- **Global e-ticaret uygulaması** → gecikmeye duyarlı
- **Aylık büyük raporlar** çalıştırılıyor

- CloudWatch'ta:
  - **ReadIOPS spike**
  - **CPUUtilization spike**

👉 Bu, raporların **okuma ağırlıklı ve ağır sorgular** olduğunu gösterir.

### 📌 **Kritik Sınav İpuçları**

Sorudaki anahtar ifadeler:

- *large reports*
- *monthly*
- *ReadIOPS + CPU spikes*
- *MOST cost-effective*

📌 Amaç:

**Üretim (primary) veritabanını rahatlatmak, ek maliyeti minimumda tutmak.**

**Sonuç Analizi:**

### ✓ **B. Migrate the monthly reporting to an Aurora Replica**

#### 🧠 **Doğru Mimari Mantığı**

- Aurora:
  - **Read Replica**'lar, okuma yükünü primary'den alır
  - Replikalar **daha ucuz ve raporlama için idealdir**
- Raporlama:
  - OLTP yerine **replica üzerinde** çalıştırılmalı

### ✓ **DOĞRU CEVAP**

- Read-heavy workload'ı primary'den alır
- Uygulama performansı düzelir
- Minimal mimari değişiklik
- En düşük ek maliyet

📌 **AWS sınavlarında:**

*Reporting load on Aurora → Read Replica*

## A. Migrate monthly reporting to Amazon Redshift

- Redshift → **OLAP / data warehouse**
- Mimari değişiklik büyük
- ETL, data sync maliyeti
- **Aylık raporlar için aşırı pahalı ve karmaşık**

## Cost-effective değil

## C. Migrate the Aurora database to a larger instance class

- Tüm workload'u büyütür
- Sürekli daha pahalı
- Sorunun kökünü çözmez (raporlar yine primary'de)

## D. Increase Provisioned IOPS

- Daha fazla disk I/O
- CPU spike devam eder
- Sürekli maliyet artışı
- Aurora zaten storage katmanında optimize

## Sonuç

- **Aurora + reporting → Read Replica**
  - **ReadIOPS spike** → okuma yükü
  - **Cost-effective** → replica > scale up
  - **Redshift** → ancak büyük analitik ihtiyacı varsa
- ◆ Aurora Ölçekleme Stratejileri

### Senaryo      Doğru Ölçekleme

Read-heavy **Scale-out (Read Replica)**

Write-heavy Scale-up (bigger instance)

Reporting      Read Replica

Analytics      Redshift (gerekliyorsa)

## SINAV ALTIN KURALLARI

- **Reporting OLTP DB'yi yoruyorsa** → Read Replica
  - **ReadIOPS spike** → Okuma ölçekte
  - **Cost-effective** → Scale-out > Scale-up
  - **Redshift** → OLAP için, hemen seçme
- 

### QUESTION 15

A company hosts a website analytics application on a single Amazon EC2 On-Demand Instance. The analytics software is written in PHP and uses a MySQL database. The analytics software, the web server that provides PHP, and the database server are all hosted on the EC2 instance. The application is showing signs of performance degradation during busy times and is presenting 5xx errors. The company needs to make the application scale seamlessly.

Which solution will meet these requirements MOST cost-effectively?

- A. Migrate the database to an Amazon RDS for MySQL DB instance. Create an AMI of the web application. Use the AMI to launch a second EC2 On-Demand Instance. Use an Application Load Balancer to distribute the load to each EC2 instance.
- B. Migrate the database to an Amazon RDS for MySQL DB instance. Create an AMI of the web application. Use the AMI to launch a second EC2 On-Demand Instance. Use Amazon Route 53 weighted routing to distribute the load across the two EC2 instances.
- C. Migrate the database to an Amazon Aurora MySQL DB instance. Create an AWS Lambda function to stop the EC2 instance and change the instance type. Create an Amazon CloudWatch alarm to invoke the Lambda function when CPU utilization surpasses 75%.
- D. Migrate the database to an Amazon Aurora MySQL DB instance. Create an AMI of the web application. Apply the AMI to a launch template. Create an Auto Scaling group with the launch template. Configure the launch template to use a Spot Fleet. Attach an Application Load Balancer to the Auto Scaling group.

**Soru:**

Bir şirket, bir **Amazon EC2 On-Demand Instance** üzerinde barındırılan bir **web sitesi analiz uygulaması** çalıştırmaktadır. Analiz yazılımı **PHP** ile yazılmıştır ve **MySQL** veritabanı kullanmaktadır. Analiz yazılımı, PHP'yi sağlayan **web sunucusu** ve **veritabanı sunucusunun** tamamı **aynı EC2 instance** üzerinde barındırılmaktadır.

Uygulama, yoğun zamanlarda **performans düşüşü** belirtileri göstermekte ve **5xx hataları** üretmektedir. Şirket, uygulamanın **kesintisiz şekilde ölçeklenebilmesini** sağlamaya ihtiyaç duymaktadır.

Bu gereksinimleri **EN maliyet etkin** şekilde karşılayan çözüm hangisidir?

- A. Veritabanını **Amazon RDS for MySQL** DB instance'ına taşımak. Web uygulamasının bir **AMI**'sini oluşturmak. Bu AMI'yi kullanarak ikinci bir **EC2 On-Demand Instance** başlatmak. Yükü her bir EC2 instance'a dağıtmak için bir **Application Load Balancer** kullanmak.
- B. Veritabanını **Amazon RDS for MySQL** DB instance'ına taşımak. Web uygulamasının bir **AMI**'sini oluşturmak. Bu AMI'yi kullanarak ikinci bir **EC2 On-Demand Instance** başlatmak. Yükü iki EC2 instance arasında dağıtmak için **Amazon Route 53 weighted routing** kullanmak.
- C. Veritabanını **Amazon Aurora MySQL** DB instance'ına taşımak. EC2 instance'ını durdurup instance tipini değiştiren bir **AWS Lambda fonksiyonu** oluşturmak. CPU kullanımı %75'i geçtiğinde Lambda fonksiyonunu tetiklemek için bir **Amazon CloudWatch alarmı** oluşturmak.
- D. Veritabanını **Amazon Aurora MySQL** DB instance'ına taşımak. Web uygulamasının bir **AMI**'sini oluşturmak. AMI'yi bir **launch template**'e uygulamak. Bu launch template ile bir **Auto Scaling group** oluşturmak. Launch template'i **Spot Fleet** kullanacak şekilde yapılandırmak. **Application Load Balancer**'ı Auto Scaling grubuna bağlamak.

**Soru Analizi:**

#### **Mevcut Mimari**

- **Tek EC2 On-Demand instance**
  - PHP web uygulaması
  - MySQL veritabanı
  - Web server + DB **aynı makinede**
- Yoğun zamanlarda:
  - **Performans düşüyor**
  - **5xx hataları** oluşuyor

#### **Gereksinimler**

Sorudaki kritik ifadeler:

1. **Scale seamlessly (kesintisiz ölçeklenme)**
2. **5xx errors** → uygulama aşırı yük altında

### 3. MOST cost-effective

4. Mevcut PHP uygulaması → korunmalı

📌 Bu, klasik bir:

**Monolithic web app → scalable web architecture** sorusudur.

**Sonuç Analizi:**

A. RDS for MySQL + ALB + ikinci EC2

 **Doğru Mimari Yaklaşım**

En iyi pratik (AWS Well-Architected):

- **Database'i ayıร → Managed DB (RDS)**
- **Web tier'i ölçekle**
  - ALB + EC2
- **Basit ve maliyet etkin**
- **Aşırı karmaşıklıktan kaçın**

**DOĞRU CEVAP**

- RDS:
  - DB yükünü EC2'den alır
  - Managed service → düşük ops yükü
- ALB:
  - HTTP/HTTPS için ideal
  - Health check + load balancing
- EC2 On-Demand:
  - Basit
  - Maliyet etkin
- Mimari:
  - Web tier yatay ölçeklenir
  - DB ayrı katman

📌 **Sınav mantığı:**

*Simple, scalable, cost-effective → ALB + EC2 + RDS*

## B. RDS + Route 53 weighted routing

- Route 53:
  - DNS seviyesinde
  - Health check & session handling zayıf
- Load balancing için **ALB varken tercih edilmez**
- Gerçek zamanlı trafik dağıtıımı yok

## C. Aurora + Lambda ile instance type değiştirme

- Instance type değiştirmek:
  - **Scale up**, scale out değil
  - Kesintili
- Lambda ile stop/start:
  - Over-engineering
  - Yavaş ve riskli
- Aurora:
  - Gereksiz pahalı

## D. Aurora + ASG + Spot Fleet + ALB

- Teknik olarak ölçülenir ama:
  - **Spot instance'lar kesilebilir**
  - Web uygulamaları için riskli
- Aurora:
  - Ek maliyet
- Soru **MOST cost-effective + simple** diyor

### Sınav tuzağı:

Çok “enterprise” çözüm → gerekenden pahalı

### Sonuç

- **Single EC2 web app → ALB + EC2**
- **DB aynı makinedeyse → RDS'ye ayır**
- **Route 53 ≠ load balancer**

- **Scale seamlessly → Scale out (ALB + EC2)**

◆ **Problem → Doğru Mimari**

Sorudaki Belirti	Anlamı	Doğru Çözüm
Single EC2	SPOF	Web tier'i ayır
5xx errors	Aşırı yük	Load balancing
Performance degradation	Ölçekleme ihtiyacı	Scale out
PHP + MySQL same host	Resource contention	DB'yi ayır (RDS)
Seamless scaling	Kesintisiz büyümeye	ALB + EC2

◆ **Servis Seçim Rehberi**

Katman	Doğru Servis	Neden
Web	<b>EC2 (AMI)</b>	Mevcut kod korunur
Load Balancing	<b>ALB</b>	HTTP/HTTPS, health check
Database	<b>RDS for MySQL</b>	Managed, ucuz
Scaling	Manuel / basit	En düşük maliyet

🔥 **SINAV ALTIN KURALLARI**

- **Web app scale** → ALB + EC2
- **DB aynı makinede** → RDS'ye ayır
- **Route 53 ≠ load balancer**
- **Aurora** → gerekmedikçe pahalı

**QUESTION 16**

A company runs a stateless web application in production on a group of Amazon EC2 On-Demand Instances behind an Application Load Balancer. The application experiences heavy usage during an 8-hour period each business day. Application usage is moderate and steady overnight. Application usage is low during weekends. The company wants to minimize its EC2 costs without affecting the availability of the application.

Which solution will meet these requirements?

- A. Use Spot Instances for the entire workload.
- B. Use Reserved Instances for the baseline level of usage. Use Spot instances for any additional capacity that the application needs.
- C. Use On-Demand Instances for the baseline level of usage. Use Spot Instances for any additional capacity that the application needs.
- D. Use Dedicated Instances for the baseline level of usage. Use On-Demand Instances for any additional capacity that the application needs.

**Soru:**

Bir şirket, **durumsuz (stateless) bir web uygulamasını** üretim ortamında **Application Load Balancer** arkasında çalışan bir grup **Amazon EC2 On-Demand Instance** üzerinde çalıştırmaktadır. Uygulama, **her iş günü boyunca 8 saatlik bir süre içinde yoğun kullanım** yaşamaktadır. **Gece boyunca** uygulama kullanımı **orta düzeyde ve stabildir**. **Hafta sonları** ise uygulama kullanımı **düşüktür**.

Şirket, uygulamanın **erişilebilirliğini etkilemeden EC2 maliyetlerini en aza indirmek** istemektedir.

Bu gereksinimleri karşılayan çözüm hangisidir?

- A. Tüm iş yükü için **Spot Instance** kullanmak.
- B. **Temel (baseline) kullanım seviyesi** için **Reserved Instance** kullanmak. Uygulamanın ihtiyaç duyduğu **ek kapasite** için **Spot Instance** kullanmak.
- C. **Temel (baseline) kullanım seviyesi** için **On-Demand Instance** kullanmak.  
Uygulamanın ihtiyaç duyduğu **ek kapasite** için **Spot Instance** kullanmak.
- D. **Temel (baseline) kullanım seviyesi** için **Dedicated Instance** kullanmak.  
Uygulamanın ihtiyaç duyduğu **ek kapasite** için **On-Demand Instance** kullanmak.

**Soru Analizi:**

 **Mevcut Durum**

- **Stateless web application**
- **ALB arkasında EC2**
- **Kullanım paterni net:**
  - İş günleri **8 saat yoğun**
  - Gece **orta ve stabil**
  - Hafta sonu **düşük**

 **Kritik Gereksinimler**

Sorudaki anahtar ifadeler:

1. **Minimize EC2 costs**
2. **Availability etkilenmemeli**
3. **Predictable baseline usage**
4. **Stateless**

💡 Bu, klasik bir:

**Baseline + burst** kapasite yönetimi sorusudur.

🧠 **Doğru Mimari Mantığı**

- **Baseline (her zaman gereklili kapasite):**
  - Kesintisiz çalışmalı
  - **Predictable**
- **Burst (yüksek saatler):**
  - Kısa süreli
  - Daha ucuz seçenekler kullanılabilir

AWS en iyi pratiği:

**Baseline = On-Demand / Reserved**

**Burst = Spot**

**Sonuç Analizi:**

✓ **C. Use On-Demand for baseline, Spot for additional capacity**

✓ **DOĞRU CEVAP**

- On-Demand:
  - Esnek
  - Availability garantili
- Spot:
  - Ucuz
  - Burst kapasite için ideal
- ALB + stateless:
  - Spot kesilirse trafik yeniden yönlendir

→ En iyi cost/availability dengesi

### ✗ A. Use Spot Instances for the entire workload

- Spot:
  - Her an kesilebilir ✗
- Availability etkilenir
  - Yanlış

### ✗ B. Use Reserved Instances for the baseline level of usage. Use Spot instances for any additional capacity

- Reserved Instances:
  - Uzun vadeli taahhüt
- Soruda:
  - Gece + hafta sonu düşük kullanım
  - Baseline değişken
- RI:
  - Gereksiz maliyet
  - En maliyet etkin değil

✖ RI, 24/7 steady workload için idealdir.

### ✗ D. Dedicated Instances for baseline

- Dedicated:
  - En pahalı seçenek ✗
- Gereksiz
  - Yanlış

### ⌚ Sonuç

- Stateless + ALB → Spot uygundur (burst)
  - Baseline predictable ama değişken → On-Demand
  - RI → 24/7 steady workload
  - Dedicated → compliance gerekirse
- ◆ Kullanım Deseni → Satın Alma Modeli

Kullanım Tipi	Sorudaki İpucu	En Uygun EC2 Seçeneği
Baseline (her zaman gerekli)	Steady / availability önemli	<b>On-Demand</b>
Burst (yoğun saatler)	Peak / heavy usage	<b>Spot Instance</b>
24/7 sabit yük	Always on	Reserved Instance
Compliance / isolation	Dedicated	Dedicated Instance

◆ **Seçenek Karşılaştırması**

Seçenek	Cost	Availability Sınav Kararı	
Spot only	★★★★★ ✗		✗
RI + Spot	★★★	✓	⚠
<b>On-Demand + Spot</b>	★★★★ ✗	✓	✓
Dedicated + On-Demand	★	✓	✗

🔥 **SINAV ALTIN KURALLARI**

- **Availability etkilenmemeli** → Spot tek başına olmaz
- **Burst usage** → Spot
- **Baseline değişken** → On-Demand
- **RI** → sadece 24/7 steady ise

### QUESTION 17

A company needs to retain application log files for a critical application for 10 years. The application team regularly accesses logs from the past month for troubleshooting, but logs older than 1 month are rarely accessed. The application generates more than 10 TB of logs per month.

Which storage option meets these requirements MOST cost-effectively?

- Store the logs in Amazon S3. Use AWS Backup to move logs more than 1 month old to S3 Glacier Deep Archive.
- Store the logs in Amazon S3. Use S3 Lifecycle policies to move logs more than 1 month old to S3 Glacier Deep Archive.

C. Store the logs in Amazon CloudWatch Logs. Use AWS Backup to move logs more than 1 month old to S3 Glacier Deep Archive.

D. Store the logs in Amazon CloudWatch Logs. Use Amazon S3 Lifecycle policies to move logs more than 1 month old to S3 Glacier Deep Archive.

**Soru:**

Bir şirketin **kritik bir uygulaması** için uygulama **log dosyalarını 10 yıl boyunca saklaması** gerekmektedir. Uygulama ekibi, **son 1 aya ait loglara** düzenli olarak **sorun giderme (troubleshooting)** amacıyla erişmektedir; ancak **1 aydan daha eski loglara** nadiren erişilmektedir. Uygulama, **ayda 10 TB'tan fazla** log üretmektedir.

Bu gereksinimleri **EN maliyet etkin** şekilde karşılayan depolama seçeneği hangisidir?

- A. Logları **Amazon S3**'te depolamak. **AWS Backup** kullanarak **1 aydan daha eski logları S3 Glacier Deep Archive**'a taşımak.
- B. Logları **Amazon S3**'te depolamak. **S3 Lifecycle policy'leri** kullanarak **1 aydan daha eski logları S3 Glacier Deep Archive**'a taşımak.
- C. Logları **Amazon CloudWatch Logs**'ta depolamak. **AWS Backup** kullanarak **1 aydan daha eski logları S3 Glacier Deep Archive**'a taşımak.
- D. Logları **Amazon CloudWatch Logs**'ta depolamak. **Amazon S3 Lifecycle policy'leri** kullanarak **1 aydan daha eski logları S3 Glacier Deep Archive**'a taşımak.

**Soru Analizi:**

 **Gereksinimler**

Sorudaki kritik ifadeler:

1. **Log retention: 10 yıl**
2. **Son 1 ay loglara sık erişim**
3. **1 aydan eski loglara nadir erişim**
4. **Ayda >10 TB log**
5. **MOST cost-effective**

 **Bu Ne Anlama Gelir?**

- **Sık erişilen veriler → S3 Standard**
- **Nadiren erişilen, uzun süre saklanan veriler → S3 Glacier Deep Archive**
- **Otomatik geçiş → S3 Lifecycle policy**
- **CloudWatch Logs:**
  - Uzun süreli saklama için **çok pahalı**

- Büyük hacimli (TB'larda) log için uygun değil

📌 AWS sınavlarında:

**Large log volume + long retention + rare access**

→ **S3 + Lifecycle + Glacier Deep Archive**

**Sonuç Analizi:**

✓ **B. S3 + S3 Lifecycle policies → Glacier Deep Archive**

🧠 **Doğru Mimari Mantığı**

- Logları **Amazon S3**'te sakla
- **Lifecycle policy** ile:
  - 30 gün → S3 Standard
  - 30 günden sonra → **S3 Glacier Deep Archive**
- 10 yıl boyunca en düşük maliyet

✓ **DOĞRU CEVAP**

- S3:
  - Büyük veri için ucuz ve ölçülebilir
- Lifecycle policy:
  - Otomatik geçiş
- Glacier Deep Archive:
  - En ucuz uzun vadeli saklama
- Minimum operasyonel yük + minimum maliyet

✗ **A. S3 + AWS Backup → Glacier Deep Archive**

- AWS Backup:
  - EC2, RDS, EFS gibi kaynaklar içindir
  - **S3 nesnelerini lifecycle yerine taşımaz**
- Gereksiz ve yanlış kullanım

✗ **C. CloudWatch Logs + AWS Backup**

- CloudWatch Logs:
  - TB'larda log için **çok pahalı**

- AWS Backup:
  - CloudWatch Logs için uygun değil  
➡ Yanlış

## ✖ D. CloudWatch Logs + S3 Lifecycle

- S3 Lifecycle:
  - CloudWatch Logs'a uygulanamaz  
➡ Yanlış

### 🎯 Sonuç

- **Long-term log retention** → S3
  - **Rare access** → Glacier Deep Archive
  - **Automatic tiering** → S3 Lifecycle
  - **CloudWatch Logs** → kısa süreli / analiz
- ◆ **Gereksinim → Doğru Servis**

Sorudaki İpucu	Ne Anlama Gelir	En Uygun Çözüm
----------------	-----------------	----------------

10 yıl saklama	Long-term retention	<b>S3</b>
Son 1 ay sık erişim	Frequent access	<b>S3 Standard</b>
1 aydan sonra nadir erişim	Archive	<b>Glacier Deep Archive</b>
>10 TB / ay	Büyük hacim	<b>S3 (CloudWatch değil)</b>
MOST cost-effective	Minimum maliyet	<b>Lifecycle policy</b>

◆ **S3 Storage Class Ezberi**

Storage Class	Ne Zaman Kullanılır
S3 Standard	Aktif / son günler
S3 Standard-IA	Nadiren erişim
S3 Glacier Flexible	Arşiv

### **S3 Glacier Deep Archive En ucuz, uzun vadeli**

### 🔥 SINAV ALTIN KURALLARI

- **Log + yıllarca saklama** → S3

- **Rare access** → Glacier Deep Archive
  - **Automation** → Lifecycle
  - **CloudWatch Logs** ≠ archive storage
- 

## QUESTION 18

A company has a data ingestion workflow that includes the following components: An Amazon Simple Notification Service (Amazon SNS) topic that receives notifications about new data deliveries An AWS Lambda function that processes and stores the data. The ingestion workflow occasionally fails because of network connectivity issues.

When failure occurs, the corresponding data is not ingested unless the company manually reruns the job.

What should a solutions architect do to ensure that all notifications are eventually processed?

- A. Configure the Lambda function for deployment across multiple Availability Zones.
- B. Modify the Lambda function's configuration to increase the CPU and memory allocations for the function.
- C. Configure the SNS topic's retry strategy to increase both the number of retries and the wait time between retries.
- D. Configure an Amazon Simple Queue Service (Amazon SQS) queue as the on-failure destination. Modify the Lambda function to process messages in the queue.

### Soru:

Bir şirketin aşağıdaki bileşenleri içeren bir **veri alım (data ingestion) iş akışı** bulunmaktadır:

- **Yeni veri teslimleri hakkında bildirimler alan bir Amazon Simple Notification Service (Amazon SNS) konusu**
- **Verileri işleyen ve depolayan bir AWS Lambda fonksiyonu**

Veri alım iş akışı, **ağ bağlantısı sorunları** nedeniyle **zaman zaman başarısız olmaktadır**. Bir hata oluşduğunda, ilgili veri **şirket işi manuel olarak yeniden çalıştırımadıkça sisteme alınmamaktadır**.

Bir solutions architect, **tüm bildirimlerin sonunda mutlaka işlenmesini** sağlamak için ne yapmalıdır?

- A. Lambda fonksiyonunu, **birden fazla Availability Zone'da dağıtılmak** şekilde yapılandırmak.
- B. Lambda fonksiyonunun yapılandırmasını değiştirerek **CPU ve bellek tahsislerini artırmak**.
- C. **SNS konusunun yeniden deneme (retry) stratejisini** yapılandırmak **hem yeniden deneme sayısını hem de denemeler arasındaki bekleme süresini artırmak**.
- D. **Hata durumunda hedef (on-failure destination)** olarak bir **Amazon Simple Queue Service (Amazon SQS) kuyruğu** yapılandırmak ve Lambda fonksiyonunu **kuyruktaki mesajları işleyecek şekilde** değiştirmek.

**Soru Analizi:**

#### **Mevcut Mimari**

- **SNS** → bildirim gönderiyor
- **Lambda** → veriyi işliyor
- **Zaman zaman network hataları** yaşanıyor
- Hata olunca:
  - **Mesaj kayboluyor**
  - **Manuel yeniden çalışma gerekiyor**

#### **Temel Gereksinim**

Sorudaki en kritik ifade:

**“ensure that all notifications are eventually processed”**

Bu şu anlama gelir:

- **At-least-once delivery**
- **Mesaj kaybı olmamalı**
- **Dayanıklı (durable) bir buffer** gerekli

#### AWS'de bunun klasik çözümü:

**SNS + SQS (durable queue) + Lambda**

**Sonuç Analizi:**

#### **D. SQS on-failure destination + Lambda**

#### **Doğru Mimari Mantığı**

- SNS:

- Push-based
- Mesajı tutmaz (short-lived retry)
- Lambda:
  - Network hatası varsa çalışmamaz
- SQS:
  - Mesajı **kalıcı olarak saklar**
  - Tüketici (Lambda) tekrar tekrar deneyebilir
  - **Eventually processed garantisi sağlar**

### ✓ DOĞRU CEVAP

- SQS:
  - Durable
  - Retry sınırsız (visibility timeout ile)
- Lambda:
  - Kuyruktan okuyarak işleyebilir
- Network düzelince:
  - Mesajlar işlenir
  - **Guaranteed delivery**

### ✗ A. Lambda'yı multi-AZ yapmak

- Lambda zaten:
  - AWS tarafından **multi-AZ** çalışır
- Network hatası çözülmmez
  - **Yanlış**

### ✗ B. CPU ve memory artırmak

- Performansı artırabilir
- Ama:
  - **Network connectivity problemini çözmez**
  - Mesaj kaybını engellemez
    - **Yanlış**

## C. SNS retry sayısını artırmak

- SNS retry:
  - Sınırlı süre ve deneme
- Uzun süreli outage durumunda:
  - Mesaj yine kaybolur
- **Eventually processed garantisi yok**  
 **Yanlış**

## Sonuç

- **Eventually processed** → SQS
  - **SNS tek başına yeterli değil**
  - **Network failure + retry ihtiyacı** → Queue
  - **Message durability** → SQS
- ◆ **Gereksinim → Doğru Servis**

Sorudaki İpucu	Anlamı	Doğru Çözüm
Eventually processed	Mesaj kaybolmamalı	<b>SQS</b>
Network failure	Geçici hata	<b>Queue buffer</b>
Manuel rerun istenmiyor	Otomatik retry	<b>Lambda + SQS</b>
High durability	Kalıcı saklama	<b>SQS</b>
Decoupling	Gevşek bağlı mimari	<b>SNS → SQS → Lambda</b>

### ◆ **SNS vs SQS (Sınav Kıyas Tablosu)**

Özellik	SNS	SQS
Push / Pull	Push	Pull
Mesaj tutma		
Retry süresi	Kısa	Uzun
Guaranteed delivery		
Network outage toleransı		

## SINAV ALTIN KURALLARI

- **Eventually processed** → SQS
  - **Failure handling** → Queue
  - **SNS tek başına yeterli değil**
  - **Mesaj kaybı = mimari hata**
- 

### QUESTION 19

A company has a service that produces event data. The company wants to use AWS to process the event data as it is received. The data is written in a specific order that must be maintained throughout processing. The company wants to implement a solution that minimizes operational overhead.

How should a solutions architect accomplish this?

- A. Create an Amazon Simple Queue Service (Amazon SQS) FIFO queue to hold messages. Set up an AWS Lambda function to process messages from the queue.
- B. Create an Amazon Simple Notification Service (Amazon SNS) topic to deliver notifications containing payloads to process. Configure an AWS Lambda function as a subscriber.
- C. Create an Amazon Simple Queue Service (Amazon SQS) standard queue to hold messages. Set up an AWS Lambda function to process messages from the queue independently.
- D. Create an Amazon Simple Notification Service (Amazon SNS) topic to deliver notifications containing payloads to process. Configure an Amazon Simple Queue Service (Amazon SQS) queue as a subscriber.

#### **Soru:**

Bir şirketin **olay (event) verisi üreten** bir servisi bulunmaktadır. Şirket, olay verilerini **alındığı anda** AWS kullanarak **işlemek** istemektedir. Veriler, **belirli bir sırada** yazılmaktadır ve bu sırانın **işleme boyunca korunması gerekmektedir**. Şirket, **operasyonel yükü en aza indiren** bir çözüm uygulamak istemektedir.

Bir solutions architect bu hedefe **nasıl ulaşmalıdır**?

- A. Mesajları tutmak için bir **Amazon Simple Queue Service (Amazon SQS) FIFO kuyruğu** oluşturmak. Kuyrukta mesajları işlemek için bir **AWS Lambda fonksiyonu** yapılandırmak.
- B. İşlenecek yükleri içeren bildirimleri iletmek için bir **Amazon Simple Notification**

**Service (Amazon SNS) konusu** oluşturmak. Bir **AWS Lambda fonksiyonunu abone (subscriber)** olarak yapılandırmak.

C. Mesajları tutmak için bir **Amazon Simple Queue Service (Amazon SQS) standard kuyruğu** oluşturmak. Kuyruktaki mesajları **bağımsız şekilde** işlemek için bir **AWS Lambda fonksiyonu** yapılandırmak.

D. İşlenecek yükleri içeren bildirimleri iletmek için bir **Amazon Simple Notification Service (Amazon SNS) konusu** oluşturmak. Bir **Amazon Simple Queue Service (Amazon SQS) kuyruğunu abone (subscriber)** olarak yapılandırmak.

**Soru Analizi:**

#### 📌 Temel Gereksinimler

Sorudaki kritik ifadeler:

1. **Event data as it is received**  
→ Gerçek zamanlı (near real-time) işleme
2. **Specific order must be maintained**  
→ **Sıralama (ordering)** garantisı şart
3. **Minimizes operational overhead**  
→ Yönetimi kolay, serverless çözüm

#### 📌 AWS Servis Eşleşmesi

Gereksinim	AWS Servisi
------------	-------------

Ordering guarantee	<b>SQS FIFO</b>
--------------------	-----------------

Serverless processing	<b>Lambda</b>
-----------------------	---------------

Düşük operasyonel yük Managed servis

📌 AWS'de **sıra garantisı** veren temel servis:

**SQS FIFO Queue**

**Sonuç Analizi:**

✓ A. **SQS FIFO + Lambda**

✓ **DOĞRU CEVAP**

- **SQS FIFO:**
  - Mesaj sırası korunur
  - Exactly-once processing (deduplication)

- **Lambda:**
  - Serverless
  - Otomatik ölçeklenir
- Operasyonel yük minimum
  - Tüm gereksinimleri karşılar

## ✗ B. SNS + Lambda

- SNS:
  - **Ordering garantisi yok**
- Lambda:
  - Evet, serverless
- Ancak sıra korunamaz
  - Yanlış

## ✗ C. SQS Standard + Lambda

- SQS Standard:
  - At-least-once
  - **Ordering yok**
- Bağımsız işleme:
  - Sıra tamamen bozulur
  - Yanlış

## ✗ D. SNS + SQS

- SNS:
  - Ordering yok
- SQS (standard varsayırlı):
  - Ordering yok
- Ayrıca:
  - İşleme kısmı eksik (Lambda yok)
  - Yanlış

## 🎯 Sonuç

- **Order must be maintained → SQS FIFO**

- Event processing + low ops → Lambda
- SNS ≠ ordering
- Standard queue ≠ ordering
- ◆ Gereksinim → Doğru Servis

Sorudaki İpucu	Anlamı	Doğru AWS Servisi
Event data as received	Gerçek zamanlı	Lambda
Order must be maintained	Sıra korunmalı	<b>SQS FIFO</b>
Exactly-once	Tekrarsız işleme	<b>SQS FIFO</b>
Min ops overhead	Serverless	Lambda

◆ Servis Karşılaştırması (Sınav Gözüyle)

Servis	Ordering Ops Yükü	Sınavda
<b>SQS FIFO</b>	✓	Düşük ✓
SQS Standard	✗	Düşük ✗
SNS	✗	Çok düşük ✗
Kinesis	✓	Orta !

### 🔥 SINAV ALTIN KURALLARI

- Order kelimesi geçiyorsa → FIFO
- Min ops → Serverless
- Event stream ama basit → SQS FIFO
- SNS = broadcast, sıra yok

### QUESTION 20

A company is migrating an application from on-premises servers to Amazon EC2 instances. As part of the migration design requirements, a solutions architect must implement infrastructure metric alarms. The company does not need to take action if CPU utilization increases to more than 50% for a short burst of time. However, if the CPU utilization increases to more than 50% and read IOPS on the disk are high at the same

time, the company needs to act as soon as possible. The solutions architect also must reduce false alarms.

What should the solutions architect do to meet these requirements?

- A. Create Amazon CloudWatch composite alarms where possible.
- B. Create Amazon CloudWatch dashboards to visualize the metrics and react to issues quickly.
- C. Create Amazon CloudWatch Synthetics canaries to monitor the application and raise an alarm.
- D. Create single Amazon CloudWatch metric alarms with multiple metric thresholds where possible.

#### Soru:

Bir şirket, bir uygulamayı şirket içi (on-premises) sunuculardan Amazon EC2 instance'larına taşıyor. Taşıma sürecinin bir parçası olarak, bir solutions architect altyapı metrikleri içinalarmlar oluşturmalıdır. Şirket, CPU kullanımının kısa süreli olarak %50'nin üzerine çıkması durumunda herhangi bir aksiyon almak istememektedir. Ancak CPU kullanımı %50'nin üzerine çıktıığında **aynı anda disk okuma IOPS değerleri de yüksekse**, şirketin mümkün olan en kısa sürede aksiyon alması gerekmektedir.

Ayrıca solutions architect, **yanlış (false)alarları azaltmak** zorundadır.

Bu gereksinimleri karşılamak için solutions architect ne yapmalıdır?

- A. Mümkün olan yerlerde Amazon CloudWatch compositealarmlar oluşturmalıdır.
- B. Metrikleri görselleştirmek sorunlara hızlıca tepki verebilmek için Amazon CloudWatch dashboard'ları oluşturmalıdır.
- C. Uygulamayı izlemek ve alarm oluşturmak için Amazon CloudWatch Synthetics canary'leri oluşturmalıdır.
- D. Mümkün olan yerlerde birden fazla metrik eşiği içeren tek Amazon CloudWatch metricaları oluşturmalıdır.

#### Soru Analizi:

Soruda özellikle vurgulanan **ana gereksinimler**:

1. **Kısa süreli CPU artışlarında aksiyon alınmamalı**  
→ Yani **tek başına CPU alarmı** istenmiyor.
2. **CPU > %50 VE aynı anda disk Read IOPS yüksekse**  
→ **Birden fazla metriğin birlikte değerlendirilmesi** gerekiyor.
3. **Mümkün olan en kısa sürede aksiyon alınmalı**  
→ Gerçek problem anında alarm hemen tetiklenmeli.

#### 4. False alarm'lar azaltılmalı

→ Gereksiz alarmlardan kaçınılmalı.

📌 Bu maddeler bize şunu söylüyor:

**Tek bir metrik yeterli değil, birden fazla alarm koşulu birlikte sağlanmalı.**

**Sonuç Analizi:**

#### A. Amazon CloudWatch composite alarms oluşturmak ✓

- Composite alarm:
  - Birden fazla CloudWatch alarmını **AND / OR mantığıyla** birleştirir.
- Bu senaryoda:
  - CPU > %50 alarmı
  - Disk Read IOPS yüksek alarmı
  - **AND koşulu ile birleştirilir**
- Sonuç:
  - CPU tek başına artarsa alarm çalmaz
  - İki metrik birlikte artarsa alarm çalar
  - False alarm'lar azalır

✓ Tüm gereksinimleri karşılar

#### B. Amazon CloudWatch dashboards ✗

- Dashboard'lar:
  - Sadece **görselleştirme** sağlar
  - Otomatik alarm veya koşul tetiklemez
- İnsan müdahalesi gerekir

✗ Alarm gereksinimini karşılamaz

#### C. CloudWatch Synthetics canaries ✗

- Canary'ler:
  - Uygulamanın **endpoint, latency, availability** durumunu test eder
- Altyapı metrik korelasyonu (CPU + IOPS) için uygun değildir

✗ Yanlış servis kullanımı

#### D. Tek metric alarmında birden fazla threshold ✗

- CloudWatch metric alarm:
  - **Sadece tek metrik** için çalışır
- Bir alarm içinde:
  - CPU + IOPS birlikte değerlendirilemez

✗ Çoklu metrik şartını karşılamaz

#### 🎯 Sonuç

Soruda aranan anahtar kelimeler:

- “**same time**” (**aynı anda**)
- “**reduce false alarms**”
- “**multiple metrics**”

📌 Bu ifadeler **Composite Alarm** işaretidir.

#### 1 “Aynı anda”, “birlikte”, “ikisi de olursa”

👉 Anahtar kelimeler:

- *at the same time*
- *if X and Y occur*
- *multiple conditions*

📌 **Doğrudan Composite Alarm** düşün.

#### 2 “False alarm’ları azaltmak”

👉 AWS genelde şu çözümleri bekler:

- Composite Alarms
- Alarm aggregation
- AND logic

❗ Tek metrik alarm = genelde **daha fazla false alarm**

#### 3 Dashboard ≠ Alarm

Sınav tuzağıdır ⚠

- Dashboard → Görselleştirme

- Alarm → Otomatik aksiyon

📌 “React quickly” yazsa bile **dashboard çoğu zaman yanlış sık.**

#### 4 Synthetics Canary ne zaman kullanılır?

Sadece şunlar için:

- Website erişilebilir mi?
- API çalışıyor mu?
- Response time iyi mi?

✗ CPU, disk, network gibi altyapı metrikleri için **kullanılmaz**.

#### 5 Metric Alarm sınırı

- **1 alarm = 1 metrik**
- Çoklu metrik = ✗
- Çoklu alarm + mantık = ✓ Composite

#### 6 Kısa süreli spike'lar

Soruda şu varsa:

- *short burst*
- *temporary increase*
- *brief spike*

📌 Tek başına alarm **istenmiyor**, mutlaka ek koşul vardır.

#### 🧠 Hızlı Ezber Cümlesi (Sınavda Akılda Kalsın)

“Birden fazla metrik + aynı anda + false alarm azaltma = Composite Alarm”

---

### QUESTION 21

A company wants to migrate its on-premises data center to AWS. According to the company's compliance requirements, the company can use only the ap-northeast-3 Region. Company administrators are not permitted to connect VPCs to the internet.

Which solutions will meet these requirements? (Choose two.)

- A. Use AWS Control Tower to implement data residency guardrails to deny internet access and deny access to all AWS Regions except ap-northeast-3.

- B. Use rules in AWS WAF to prevent internet access. Deny access to all AWS Regions except ap-northeast-3 in the AWS account settings.
- C. Use AWS Organizations to con gure service control policies (SCPS) that prevent VPCs from gaining internet access. Deny access to all AWS Regions except ap-northeast-3.
- D. Create an outbound rule for the network ACL in each VPC to deny all traffic from 0.0.0.0/0. Create an IAM policy for each user to prevent the use of any AWS Region other than ap-northeast-3.
- E. Use AWS Config to activate managed rules to detect and alert for internet gateways and to detect and alert for new resources deployed outside of ap-northeast-3.

**Soru:**

Bir şirket, şirket içi (on-premises) veri merkezini AWS'e taşımak istiyor. Şirketin uyumluluk (compliance) gereksinimlerine göre **yalnızca ap-northeast-3 Bölgesi** kullanılabilir. Şirket yöneticilerinin **VPC'leri interne ba glamasına izin verilmemektedir.**

**Hangi çözümler bu gereksinimleri karşılar? (İki tanesini seçin.)**

- A. Internet erişimini reddeden ve ap-northeast-3 dışındaki tüm AWS Bölgelerine erişimi engelleyen veri yerleşimi (data residency) guardrail'lerini uygulamak için **AWS Control Tower** kullanın.
- B. Internet erişimini engellemek için **AWS WAF** kuralları kullanın. AWS hesap ayarlarında ap-northeast-3 dışındaki tüm AWS Bölgelerine erişimi reddedin.
- C. **AWS Organizations** kullanarak, VPC'lerin internet erişimi kazanmasını engelleyen **Service Control Policy (SCP)**'ler yapılandırın. ap-northeast-3 dışındaki tüm AWS Bölgelerine erişimi reddedin.
- D. Her VPC'de, **0.0.0.0/0** adresinden gelen tüm trafiği reddeden bir **network ACL outbound kuralı** oluşturun. Her kullanıcı için ap-northeast-3 dışındaki herhangi bir AWS Bölgesinin kullanılmasını engelleyen bir **IAM politikası** oluşturun.
- E. Internet gateway'lerini tespit edip uyarı veren ve ap-northeast-3 dışında oluşturulan yeni kaynakları tespit edip uyarı veren **AWS Config managed rules**'ları etkinleştirin.

**Soru Analizi:**

Soruda **iki ana zorunluluk** var:

**Gereksinim 1: Bölge (Region) kısıtı**

- **Sadece ap-northeast-3** kullanılabilir
- Diğer tüm AWS Region'ları **engellenmeli**

- Bu, **önleyici (preventive)** bir kontrol olmalı

Yani “sonradan uyarı” değil, **başından engelleme**

### Gereksinim 2: İnternet erişimi yasağı

- **VPC’ler internete bağlanamaz**
- Internet Gateway, NAT Gateway vb. **oluşturulamamalı veya kullanılamamalı**
- Yine **önleyici kontrol** gereklidir

 Soruda “detect”, “alert” gibi ifadeler geçmiyor → **uyarı yeterli değil**

### Kritik AWS Kavramları (Sınav İpucu)

Hizmet	Ne yapar	Önleyici mi?
<b>SCP (Service Control Policy)</b>	Hesap seviyesinde <i>maksimum yetkiyi</i> sınırlar	 Evet
<b>IAM Policy</b>	Kullanıcı/rol bazlı	 Kolayca aşılabilir
<b>AWS Config</b>	Tespit & raporlama	 Hayır
<b>AWS WAF</b>	HTTP/HTTPS trafiği	 VPC internetini kesmez
<b>Network ACL</b>	Ağ seviyesinde trafik	 Yönetimsel kısıt koymaz

### Sonuç Analizi:

#### ◆ A. AWS Control Tower + guardrails

İnterneti reddeden ve ap-northeast-3 dışını engelleyen guardrail’ler

### Artıları

- Control Tower guardrail’leri **SCP tabanlıdır**
- Region kısıtlaması yapılabılır
- Merkezi ve önleyici kontrol sağlar

### Ama:

- Control Tower, **zorunlu değildir** ve soruda özellikle geçmiyor
- İnternet erişimini VPC seviyesinde **doğrudan garanti altına almaz**

 Sınav bakış açısı:

- **Olası ama daha dolaylı** bir çözüm
- SCP'yi doğrudan kullanan seçenek varken genelde ikinci planda kalır

 **Genelde doğru kabul edilmez**

◆ **C. AWS Organizations + SCP**

VPC'lerin internet erişimini engelleyen SCP  
ap-northeast-3 dışındaki tüm region'ları reddeden SCP

 **KESİN DOĞRU**

**Neden?**

- SCP:
  - Tüm hesaplar için **üst sınır** koyar
  - IAM ile **aşılamaz**
- Sunular engellenebilir:
  - ec2:CreateInternetGateway
  - ec2:AttachInternetGateway
  - ec2:CreateNatGateway
- Ayrıca:
  - aws:RequestedRegion condition ile
  - ap-northeast-3 dışı **tamamen kapatılabilir**

**Sınavın aradığı “en doğru” çözüm**

◆ **B. AWS WAF + account region ayarları**

 **Yanlış**

**Neden?**

- AWS WAF:
  - Sadece **HTTP/HTTPS** trafiğini kontrol eder
  - Internet Gateway, NAT Gateway gibi kaynakları **engellemez**
- “AWS account settings” diye **region deny mekanizması yok**
  - Region kısıtı **SCP ile yapılır**

🚫 Bu seçenek kavramsal olarak hatalı

◆ D. Network ACL + IAM policy

✗ Yanlış

Neden?

- Network ACL:
  - Trafiği keser ama
  - Internet Gateway oluşturulmasını **engellemez**
- IAM policy:
  - Kullanıcı bazlıdır
  - Root veya farklı rol ile **bypass edilebilir**
- Merkezi, zorunlu bir kontrol değildir

🚫 Compliance için **yetersiz**

◆ E. AWS Config rules (detect & alert)

✗ Yanlış

Neden?

- AWS Config:
  - **Sadece tespit eder**
  - Kaynak oluşturulmasını **engellemez**
- Soruda:

“can use only ap-northeast-3”

“are not permitted”

➡ Bu ifadeler **önleyici kontrol** gerektirir

🎯 Sonuç

**Nihai Doğru Cevaplar**

**Doğru seçenekler:**

C

A (*ikincil ama kabul edilen çözüm olarak*)

Ancak sınav mantığında **en güçlü ve net çözüm kesinlikle C'dir.**

## Sınavda Hatırlaman Gereken Altın Kural 🧠

🔒 Compliance + Region restriction + Internet ban = SCP

AWS sınavlarında:

- Prevent ≠ Detect
  - Organization-level ≠ User-level
- 

## QUESTION 22

A company uses a three-tier web application to provide training to new employees. The application is accessed for only 12 hours every day. The company is using an Amazon RDS for MySQL DB instance to store information and wants to minimize costs.

What should a solutions architect do to meet these requirements?

- A. Configure an IAM policy for AWS Systems Manager Session Manager. Create an IAM role for the policy. Update the trust relationship of the role. Set up automatic start and stop for the DB instance.
- B. Create an Amazon ElastiCache for Redis cache cluster that gives users the ability to access the data from the cache when the DB instance is stopped. Invalidate the cache after the DB instance is started.
- C. Launch an Amazon EC2 instance. Create an IAM role that grants access to Amazon RDS. Attach the role to the EC2 instance. Configure a cron job to start and stop the EC2 instance on the desired schedule.
- D. Create AWS Lambda functions to start and stop the DB instance. Create Amazon EventBridge (Amazon CloudWatch Events) scheduled rules to invoke the Lambda functions. Configure the Lambda functions as event targets for the rules.

### Soru:

Bir şirket, yeni çalışanlara eğitim vermek için **üç katmanlı (three-tier) bir web uygulaması** kullanmaktadır. Uygulamaya **günde yalnızca 12 saat** erişilmektedir. Şirket, bilgileri depolamak için **Amazon RDS for MySQL DB instance** kullanmaktadır ve **maliyetleri en aza indirmek** istemektedir.

Bu gereksinimleri karşılamak için bir **solutions architect** ne yapmalıdır?

- A. AWS Systems Manager **Session Manager** için bir **IAM policy** yapılandırın. Bu politika için bir **IAM role** oluşturun. Rolün **trust relationship**'ini güncelleyin. DB instance için **otomatik başlatma ve durdurma** ayarlayın.

**B.** DB instance durdurulduğunda kullanıcıların verilere cache üzerinden erişebilmesini sağlayan bir **Amazon ElastiCache for Redis cache cluster** oluşturun. DB instance başlatıldıkten sonra cache'i geçersiz kıllın (invalidate).

**C.** Bir **Amazon EC2 instance** başlatın. Amazon RDS'ye erişim izni veren bir **IAM role** oluşturun ve bu rolü EC2 instance'a ekleyin. İstenen zaman çizelgesine göre EC2 instance'i başlatıp durdurmak için bir **cron job** yapılandırın.

**D.** DB instance'ı başlatmak ve durdurmak için **AWS Lambda function'ları** oluşturun. Bu Lambda function'ları tetiklemek için **Amazon EventBridge (Amazon CloudWatch Events)** zamanlanmış kuralları oluşturun. Lambda function'ları bu kuralların **event target**'ı olarak yapılandırın.

#### Soru Analizi:

Soruda gizli ama çok net **3 kritik ipucu** var:

- 1. Uygulama günde sadece 12 saat kullanılıyor**
- 24/7 çalışması gereksiz**
- 2. Amazon RDS for MySQL kullanılıyor**
- RDS durdurulup başlatılabilir** (stop/start desteklenir)
- 3. Maliyetleri en aza indirmek isteniyor**
- Kullanılmadığı saatlerde **DB kapalı olmalı**
- Otomasyon şart (manuel kabul edilmez)

#### En Uygun Maliyet Optimizasyonu Nedir?

##### **RDS stop/start + zamanlama**

- RDS durdurulduğunda:
  - Compute ücreti **X**
  - Storage ücreti **✓** (ödenir)
- Eğitim uygulaması için **tam ideal çözüm**

#### Sonuç Analizi:

##### **◆ D. Lambda + EventBridge (CloudWatch Events)**

- DOĞRU CEVAP**

#### Neden?

- EventBridge:

- Zamanlanmış kural (cron)

- Lambda:

- StartDBInstance
- StopDBInstance

- Tamamen:

- **Serverless**
- **Otomatik**
- **En düşük maliyet**

- ◆ **A. Systems Manager Session Manager + IAM**

 Yanlış

**Neden?**

- Session Manager:
  - EC2'lere **shell erişimi** içindir
  - RDS başlatma/durdurma ile **ilgisi yok**
- IAM rol oluşturmak **tek başına bir çözüm değil**
- Otomasyon mekanizması yok

 Sınavda “alakası olmayan servis” tuzağı

- ◆ **B. ElastiCache + DB kapalıken veri erişimi**

 Yanlış

**Neden?**

- DB kapalıken:
  - Uygulama **çalışamaz**
- Cache:
  - DB'nin yerine **geçemez**
- Ek maliyet oluşturur → “minimize cost”a aykırı

 Mantıksal olarak da hatalı

- ◆ **C. EC2 + cron job**

## Yanlış

### Neden?

- EC2 başlat/durdurmak:
  - RDS maliyetini **etkilemez**
- EC2 üzerinden RDS kontrol etmek:
  - Gereksiz karmaşıklık
- EC2:
  - Ek compute maliyeti

 AWS best practice değil

## Sonuç

### AWS Exam Trick (Altın Kural)

 Belirli saatlerde çalışan sistem + cost optimization = Lambda + EventBridge

### Bu Soruyu 3 Saniyede Çözme Yöntemi

Soruda şunları görürsen:

- “only X hours a day”
- “minimize costs”
- “RDS”

 STOP / START RDS

 Lambda + EventBridge

### Neden BU çözüm en ucuz?

**Çözüm**      **Maliyet**

Lambda       yok

EventBridge       yok

EC2       yok

RDS compute  kapalıken

RDS storage       ödenir (kaçınılmaz)

### AWS Sınavında 1 Bakışta Anla

Soruda şu kelimeler varsa:

- “only X hours a day”
- “training / dev / test”
- “minimize cost”
- “RDS”

→ **Lambda + EventBridge + RDS stop/start**

**Kısa Ezber Cümlesi (Exam Gold)**

**“Zamanlı çalışan RDS + düşük maliyet = EventBridge + Lambda”**

---

### **QUESTION 23**

A company sells ringtones created from clips of popular songs. The files containing the ringtones are stored in Amazon S3 Standard and are at least 128 KB in size. The company has millions of files, but downloads are infrequent for ringtones older than 90 days. The company needs to save money on storage while keeping the most accessed files readily available for its users.

Which action should the company take to meet these requirements MOST cost-effectively?

- Configure S3 Standard-Infrequent Access (S3 Standard-IA) storage for the initial storage tier of the objects.
- Move the files to S3 Intelligent-Tiering and configure it to move objects to a less expensive storage tier after 90 days.
- Configure S3 inventory to manage objects and move them to S3 Standard-Infrequent Access (S3 Standard-1A) after 90 days.
- Implement an S3 Lifecycle policy that moves the objects from S3 Standard to S3 Standard-Infrequent Access (S3 Standard-1A) after 90 days.

#### **Soru:**

Bir şirket, popüler şarkıların kısa kliplerinden oluşturulan zil sesleri satmaktadır. Zil seslerini içeren dosyalar **Amazon S3 Standard** depolama sınıfında saklanmaktadır ve dosyaların her biri **en az 128 KB** boyutundadır. Şirketin **milyonlarca dosyası** vardır, ancak **90 günden daha eski zil sesleri nadiren indirilmektedir**. Şirket, **en çok erişilen dosyaları kullanıcılar için hızlıca erişilebilir tutarken**, depolama maliyetlerini düşürmek istemektedir.

Bu gereksinimleri **en maliyet etkin** şekilde karşılamak için şirket hangi aksiyonu almalıdır?

- A. Nesneler için başlangıç depolama katmanı olarak **S3 Standard-Infrequent Access (S3 Standard-IA)** yapılandırmak.
- C. Nesneleri yönetmek için **S3 Inventory** yapılandırmak ve nesneleri **90 gün sonra S3 Standard-Infrequent Access (S3 Standard-IA)** sınıfına taşımak.
- D. Nesneleri **S3 Standard**'dan **S3 Standard-Infrequent Access (S3 Standard-IA)** sınıfına **90 gün sonra taşıyan bir S3 Lifecycle (Yaşam Döngüsü)** politikası uygulamak.

#### Soru Analizi:

Soruda geçen **kritik ifadeleri** tek tek inceleyelim:

#### Dosyalar nerede?

- **Amazon S3 Standard**
- Başlangıçta sık erişim var

👉 Demek ki **ilk 90 gün** için S3 Standard doğru bir seçim.

#### Dosya boyutu

- **En az 128 KB**

👉 Bu önemli çünkü:

- **S3 Standard-IA** için minimum nesne boyutu **128 KB**
- Yani IA'ya geçiş teknik olarak mümkün

#### Erişim paterni

- **90 günden sonra nadiren indiriliyor**

👉 Bu tam olarak **Infrequent Access (IA)** senaryosu:

- Nadiren erişilen
- Ama gerektiğinde **hemen erişilebilir** olmalı

#### Hedef

- **Maliyeti düşürmek**
- **Sık erişilen dosyalar hızlı erişimde kalsın**

👉 Otomatik, düşük operasyonel yükü olan bir çözüm tercih edilmeli.

#### Sonuç Analizi:

**D. S3 Lifecycle policy ile 90 gün sonra IA'ya taşıma**

“Implement an S3 Lifecycle policy...”

**Neden doğru?**

- İlk 90 gün:
  - S3 Standard → hızlı ve ucuz erişim
- 90 gün sonra:
  - S3 Standard-IA → daha ucuz depolama
- **Otomatik**
- **Operasyonel yük yok**
- AWS'in önerdiği en iyi uygulama

**Maliyet + performans dengesi = mükemmel**

**A. Başlangıçta S3 Standard-IA kullanmak**

*Configure S3 Standard-Infrequent Access as the initial storage tier*

**Neden yanlış?**

- Yeni zil sesleri sık indiriliyor
- S3 Standard-IA:
  - Depolama ucuz
  - Erişim (retrieval) pahalı
- Sık erişimde toplam maliyet artar

**Sınav kuralı:**

“Frequently accessed data - Standard-IA”

**B. 90 gün sonra S3 Glacier'a taşımak**

*(Tipik sınav şıklığı)*

**Neden yanlış?**

- Glacier:
  - Arşiv amaçlı
  - Dakika–saat süren geri getirme

- Zil sesleri:
  - Kullanıcı talebiyle
  - Anında erişim gerektirir

 **Altın kural:**

"User-facing content - Glacier"

 **C. S3 Inventory ile nesneleri yönetip taşımak**

*Configure S3 Inventory and move objects after 90 days*

 **Neden yanlış?**

- S3 Inventory sadece raporlama yapar
- Nesne taşımaz
- Ek script, manuel işlem gereklidir
- Operasyonel karmaşıklık ↑

 **AWS sınavlarında:**

"Inventory - Automation"

 **Sonuç**

 **Sınav İpucu (Altın Kural)**

"Time-based access pattern + S3 = Lifecycle Policy"

**S3 Standard-IA vs S3 Glacier (Sınavda Çok Çıkar)**

Özellik	S3 Standard-IA	S3 Glacier
Erişim sıklığı	Nadiren	Çok nadiren
Erişim süresi	Anında (ms)	Dakika-saat
Retrieval ücreti	Var	Var (daha karmaşık)
Kullanım senaryosu	Yedek, eski ama aktif veri Arşiv, compliance	

Bu soruya uygun mu?  EVET

 HAYIR

 **Altın kural:**

“Immediately available” → **IA**

“Archive / long-term retention” → **Glacier**

---

## QUESTION 24

A company needs to save the results from a medical trial to an Amazon S3 repository. The repository must allow a few scientists to add new files and must restrict all other users to read-only access. No users can have the ability to modify or delete any files in the repository. The company must keep every file in the repository for a minimum of 1 year after its creation date.

Which solution will meet these requirements?

- A. Use S3 Object Lock in governance mode with a legal hold of 1 year.
- B. Use S3 Object Lock in compliance mode with a retention period of 365 days. Topic 1
- C. Use an IAM role to restrict all users from deleting or changing objects in the S3 bucket. Use an S3 bucket policy to only allow the IAM role.
- D. Configure the S3 bucket to invoke an AWS Lambda function every time an object is added. Configure the function to track the hash of the saved object so that modified objects can be marked accordingly.

### Soru:

Bir şirket, bir **tıbbi deneyin sonuçlarını** Amazon S3 üzerinde bir depoya kaydetmek istemektedir. Bu depo, **yalnızca birkaç bilim insanının yeni dosya eklemesine** izin vermelii ve **diğer tüm kullanıcılar için salt okunur (read-only) erişim** sağlamalıdır. Hiçbir kullanıcının depodaki **herhangi bir dosyayı değiştirmeye veya silmeye yetkisi olmamalıdır**. Şirket, depodaki **her dosyayı oluşturulma tarihinden itibaren en az 1 yıl boyunca saklamak** zorundadır.

Bu gereksinimleri **hangi çözüm karşıları?**

- A. 1 yıllık yasal bekletme (legal hold) ile S3 Object Lock'u governance modunda kullanmak.**
- B. 365 günlük saklama süresi ile S3 Object Lock'u compliance modunda kullanmak.**
- C. Tüm kullanıcıların S3 bucket içindeki nesneleri silmesini veya değiştirmesini engelleyen bir IAM rolü kullanmak ve yalnızca bu rolü izinli kılacak bir S3 bucket policy yapılandırmak.**
- D. Bir nesne eklendiğinde S3 bucket'ın bir AWS Lambda fonksiyonunu tetikleyecek şekilde yapılandırılması ve bu fonksiyonun kaydedilen nesnenin hash değerini takip ederek değiştirilmiş nesneleri işaretlemesi.**

## Soru Analizi:

### Veri türü: Tıbbi deney sonuçları

- Regülasyonlu veri
- Değiştirilemezlik (immutability) çok önemli

### Yetki modeli

- Az sayıda kullanıcı → dosya ekleyebilir
- Herkes → sadece okuma
- ✗ Kimse → değiştiremez / silemez

➡ Bu, klasik **WORM (Write Once, Read Many)** senaryosudur.

### Saklama süresi

- Her dosya en az 1 yıl saklanmalı
  - Oluşturuluktan sonra silinememeli
- ➡ “Minimum retention” ifadesi çok kritik.

### Seçenek Analizi:

#### B. Object Lock (Compliance mode) + 365 gün retention

*Compliance mode with a retention period of 365 days*

#### Neden doğru?

- **Compliance mode:**
  - ✗ Hiç kimse (root dahil) dosyayı:
    - silemez
    - değiştiremez
    - retention’ı kısaltamaz
- **Retention period:**
  - Minimum 1 yıl şartını garanti eder
- Tam **WORM davranışı**

➡ Medikal / finansal / yasal veriler için **AWS best practice**

#### A. Object Lock (Governance mode) + legal hold 1 yıl

*Governance mode with a legal hold of 1 year*

### Neden yanlış?

- **Governance mode:**

- Yetkili kullanıcılar **lock'u bypass edebilir**

- Soruda açıkça:

“No users can have the ability to modify or delete any files”

 Governance ≠ mutlak koruma

 Regülasyonlu veri için **yetersiz**

### C. IAM rolü + bucket policy

*Restrict users with IAM and bucket policy*

### Neden yanlış?

- IAM politikaları:

- **Değiştirilebilir**
  - Root veya admin tarafından açılabilir

- **Retention garantisı yok**

- Regülasyon gereksinimini karşılamaz

 AWS sınavında:

“Policy-based protection ≠ immutability”

### D. Lambda ile hash takibi

*Track object hash to detect modifications*

### Neden yanlış?

- Değişikliği **engellemez**, sadece **tespit eder**
- Silmeyi hiç engellemez
- Karmaşık, pahalı, hataya açık

 Sınavda:

“Detect ≠ Prevent”

### Sonuç

“Regulated data + minimum retention = S3 Object Lock (Compliance)”

## 🔥 S3 Object Lock – Governance vs Compliance (Net Karşılaştırma)

Özellik	Governance Mode	Compliance Mode
Veri silinebilir mi?	⚠ Yetkili kullanıcı bypass edebilir	✗ Kimse silemez
Retention kısaltılabilir mi?	⚠ Admin yapabilir	✗ Asla
Root kullanıcı	Bypass edebilir	✗ Bypass edemez
Regülasyon uyumu	✗ Zayıf	✓ Güçlü
Medikal / finansal veri	✗	✓
Sınavda anlamı	“Yumuşak koruma”	“Mutlak koruma”

### 📌 Altın kural:

“Minimum retention + no exceptions” → **Compliance**

### 🧠 Sınav Tuzakları (Çok Çıkar)

- ✗ IAM Policy ≠ Immutability
- ✗ Hash / audit ≠ prevention
- ✗ Governance ≠ compliance
- ✓ WORM data = Object Lock (Compliance)

---

## QUESTION 25

A large media company hosts a web application on AWS. The company wants to start caching confidential media files so that users around the world will have reliable access to the files. The content is stored in Amazon S3 buckets. The company must deliver the content quickly, regardless of where the requests originate geographically.

Which solution will meet these requirements?

- Use AWS DataSync to connect the S3 buckets to the web application.
- Deploy AWS Global Accelerator to connect the S3 buckets to the web application.
- Deploy Amazon CloudFront to connect the S3 buckets to CloudFront edge servers.
- Use Amazon Simple Queue Service (Amazon SQS) to connect the S3 buckets to the web application.

### Soru:

Büyük bir medya şirketi, AWS üzerinde barındırılan bir web uygulamasına sahiptir. Şirket, **gizli (confidential) medya dosyalarını önbelleğe almak (cache etmek)** istemektedir; böylece **dünyanın dört bir yanındaki kullanıcılar dosyalara güvenilir şekilde erişebilecektir**. İçerikler **Amazon S3 bucket'larında** saklanmaktadır. Şirket, **isteklerin coğrafi olarak nereden geldiğinden bağımsız olarak içeriği hızlı bir şekilde sunmak** zorundadır.

Bu gereksinimleri **hangi çözüm karşıları?**

- A. S3 bucket'larını web uygulamasına bağlamak için **AWS DataSync** kullanmak.
- B. S3 bucket'larını web uygulamasına bağlamak için **AWS Global Accelerator** dağıtmak.
- C. **Amazon CloudFront** dağıtarak S3 bucket'larını **CloudFront edge sunucularına** bağlamak.
- D. S3 bucket'larını web uygulamasına bağlamak için **Amazon Simple Queue Service (Amazon SQS)** kullanmak.

### Soru Analizi:

Sorudaki **ana gereksinimleri** ayıralım:

#### 1 Veri türü

- **Gizli (confidential) medya dosyaları**
- **Amazon S3'te saklanıyor**

👉 Güvenli ve kontrollü erişim şart.

#### Performans gereksinimi

- **Dünyanın her yerinden hızlı erişim**
- Coğrafyadan bağımsız düşük gecikme

👉 **Global edge cache** ihtiyacı.

#### Teknik ihtiyaç

- **Caching (önbelleklemeye)** açıkça belirtilmiş
- Sadece yönlendirme değil, içerik **kopyalanıp dağıtılmalı**

👉 CDN (Content Delivery Network) gereksinimi.

### Seçenek Analizi:

#### C. Amazon CloudFront

*Connect S3 buckets to CloudFront edge servers*

 **Neden doğru?**

- CloudFront:
  - AWS'nin **CDN servisi**
  - İçeriği **edge location'larda cache'ler**
  - Global düşük gecikme
- S3:
  - **Native origin** desteği
- Confidential content:
  - Signed URLs / Signed Cookies
  - OAI / OAC ile erişim kısıtlama

 **Caching + global delivery = CloudFront**

 **A. AWS DataSync**

*Use AWS DataSync to connect the S3 buckets to the web application*

 **Neden yanlış?**

- DataSync:
  - Veri **taşıma / senkronizasyon** servisi
  - Cache veya global dağıtım yapmaz
- Performans iyileştirme amacı yok

 **Data transfer ≠ content delivery**

 **B. AWS Global Accelerator**

*Deploy AWS Global Accelerator*

 **Neden yanlış?**

- Global Accelerator:
  - **Network routing** optimizasyonu yapar
  - Cache **yapmaz**
- S3 için doğrudan CDN çözümü değildir

📌 **Routing ≠ caching**

✗ **D. Amazon SQS**

Use Amazon SQS

✗ **Neden yanlış?**

- SQS:
  - Mesaj kuyruğu
  - Dosya dağıtımını veya cache yapmaz

📌 **Messaging ≠ content delivery**

🎯 **Sonuç**

“Global + cache + S3 = CloudFront”

🔥 **CloudFront Güvenliği: OAI vs OAC (Sınav Favorisi)**

Özellik	OAI (Origin Access Identity) OAC (Origin Access Control)
Güncel AWS önerisi	✗ Eski
Desteklenen origin	S3
Güvenlik	Orta
IAM policy entegrasyonu	Sınırlı
Sınavda tercih	Nadiren
	<b>Yeni &amp; önerilen</b>
	<b>S3 + diğer origin'ler</b>
	<b>Daha güçlü</b>
	<b>Daha iyi</b>
	<b>Sıkça doğru cevap</b>

📌 **Altın kural:**

“Secure S3 + CloudFront” → **OAC**

🧠 **CloudFront ile Gizli (Confidential) İçerik Koruma Yöntemleri**

Soruda “**confidential media files**” geçtiği için bunlar aklında olmalı:

- **Signed URLs** → tek kullanıcıya özel erişim
- **Signed Cookies** → çoklu dosya erişimi
- **OAI / OAC** → S3'e doğrudan erişimi kapatır
- **HTTPS (TLS)** → şifreli iletişim

📌 **Sınavda:**

“Confidential + CDN” → **CloudFront + Signed URL/Cookie**

💡 **CloudFront vs Global Accelerator (Çok Karışır!)**

**Özellik**      **CloudFront Global Accelerator**

Cache      Var      Yok

CDN      Evet      Hayır

S3 ile kullanım      Native      Dolaylı

Media delivery      En iyi      Uygun değil

Network routing     

📌 **Kısa ezber:**

“Dosya/medya” → CloudFront

“TCP/UDP app” → Global Accelerator

---

## QUESTION 26

A company produces batch data that comes from different databases. The company also produces live stream data from network sensors and application APIs. The company needs to consolidate all the data into one place for business analytics. The company needs to process the incoming data and then stage the data in different Amazon S3 buckets. Teams will later run one-time queries and import the data into a business intelligence tool to show key performance indicators (KPIs).

Which combination of steps will meet these requirements with the LEAST operational overhead? (Choose two.)

- A. Use Amazon Athena for one-time queries. Use Amazon QuickSight to create dashboards for KPIs.
- B. Use Amazon Kinesis Data Analytics for one-time queries. Use Amazon QuickSight to create dashboards for KPIs.
- C. Create custom AWS Lambda functions to move the individual records from the databases to an Amazon Redshift cluster.
- D. Use an AWS Glue extract, transform, and load (ETL) job to convert the data into JSON format. Load the data into multiple Amazon OpenSearch Service (Amazon Elasticsearch Service) clusters.

E. Use blueprints in AWS Lake Formation to identify the data that can be ingested into a data lake. Use AWS Glue to crawl the source, extract the data, and load the data into Amazon S3 in Apache Parquet format.

#### Soru:

Bir şirket, **farklı veritabanlarından gelen toplu (batch) veriler** üretmektedir. Şirket ayrıca **ağ sensörlerinden ve uygulama API'lerinden gelen canlı akış (live stream) verileri** de üretmektedir. Şirket, **tüm verileri iş analitiği için tek bir yerde birleştirmek** istemektedir. Gelen verilerin **işlenmesi** ve ardından verilerin **farklı Amazon S3 bucket'larına geçici olarak yerleştirilmesi (stage edilmesi)** gerekmektedir. Ekipler daha sonra **tek seferlik sorgular (one-time queries)** çalıştıracak ve verileri **temel performans göstergelerini (KPI)** göstermek için bir **iş zekâsı (BI)** aracına aktaracaktır.

Bu gereksinimleri **EN AZ operasyonel yükle** karşılayan **iki adım kombinasyonu** hangisidir? (**İki seçenek seçin.**)

- A. Tek seferlik sorgular için **Amazon Athena** kullanmak. KPI panoları (dashboard) oluşturmak için **Amazon QuickSight** kullanmak.
- B. Tek seferlik sorgular için **Amazon Kinesis Data Analytics** kullanmak. KPI panoları oluşturmak için **Amazon QuickSight** kullanmak.
- C. Veritabanlarındaki tekil kayıtları bir **Amazon Redshift** kümesine taşımak için **özel AWS Lambda fonksiyonları** oluşturmak.
- D. Veriyi **JSON** formatına **dönüştürmek** için bir **AWS Glue ETL işi** kullanmak ve veriyi birden fazla **Amazon OpenSearch Service (Amazon Elasticsearch Service)** kümesine yüklemek.
- E. Bir veri gölüne (data lake) alınabilecek verileri belirlemek için **AWS Lake Formation blueprint'lerini** kullanmak. Kaynakları taramak (crawl), veriyi çıkarmak ve **Apache Parquet** formatında **Amazon S3'e yüklemek** için **AWS Glue** kullanmak.

#### Soru Analizi:

##### Veri türleri

- **Batch data** → farklı veritabanlarından
- **Streaming data** → sensörler + API'ler

👉 Yani **data lake** yaklaşımı gerekiyor.

##### Veri akışı

- Gelen veri **işlenecek**
- Sonra **farklı S3 bucket'larında stage edilecek**

👉 Merkezi depolama: **Amazon S3**

### Analitik kullanım

- **One-time queries** (ad-hoc sorgular)
- Sonradan **BI tool** ile KPI gösterimi

👉 Sürekli çalışan cluster **istenmiyor**.

### En kritik ifade

#### LEAST operational overhead

📌 Yani:

- ✗ Custom code
- ✗ Yönetilen cluster'lar
- ✓ Serverless
- ✓ Managed services

### Seçenek Analizi:

#### ✓ A. Amazon Athena + Amazon QuickSight

*Athena for one-time queries, QuickSight for dashboards*

#### ✓ Neden doğru?

- **Athena:**
  - Serverless
  - S3 üzerindeki veriye **doğrudan soru**
  - One-time query için ideal
- **QuickSight:**
  - Fully managed BI tool
  - Athena ile native entegrasyon

#### ✗ Ad-hoc analytics + düşük operasyonel yük

#### ✓ E. Lake Formation + Glue + S3 (Parquet)

*Data lake ingestion with Glue & Parquet*

#### ✓ Neden doğru?

- **Lake Formation:**
  - Data lake yönetimini kolaylaştırır
- **AWS Glue:**
  - Serverless ETL
  - Batch + stream ingestion destekler
- **Parquet format:**
  - Columnar
  - Athena & QuickSight için optimize

### ✖ **Modern AWS data lake best practice**

## ✖ **B. Kinesis Data Analytics + QuickSight**

*Kinesis Data Analytics for one-time queries*

### ✖ **Neden yanlış?**

- Kinesis Data Analytics:
  - **Real-time stream processing**
  - One-time / ad-hoc query için uygun değil
- Sürekli çalışan uygulama gereklidir

### ✖ **Streaming ≠ ad-hoc analytics**

## ✖ **C. Lambda + Redshift**

*Custom Lambda to load Redshift*

### ✖ **Neden yanlış?**

- Redshift:
  - Cluster yönetimi gereklidir
- Lambda:
  - Custom development
- Yüksek operasyonel yük

### ✖ **LEAST overhead şartını bozuyor**

## ✖ **D. Glue ETL + OpenSearch**

*JSON + OpenSearch clusters*

### ✖ Neden yanlış?

- OpenSearch:
  - Log search / near real-time analytics
- Birden fazla cluster:
  - Yönetim maliyeti yüksek
- JSON:
  - Analitik için verimsiz

### 📌 Analytics KPI ≠ OpenSearch

### 🎯 Sonuç

“Least overhead analytics = S3 + Glue + Athena + QuickSight”

### 🔥 Athena vs Redshift vs OpenSearch (Sınav Karşılaştırması)

Özellik	Athena	Redshift	OpenSearch
Tür	Serverless query	Data warehouse	Search & log analytics
Cluster yönetimi	✖ Yok	✓ Var	✓ Var
One-time query	✓ Mükemmel	✖ Aşırı	✖ Uygun değil
S3 ile çalışma	✓ Native	⚠ COPY gereklidir	⚠
KPI / BI	✓	✓	✖
Least overhead	★★★★★	★★★	★

### 📌 Altın kural:

“Ad-hoc / one-time query” → **Athena**

### 🔥 AWS Glue vs Lambda (ETL Sınav Tuzakları)

Özellik	AWS Glue	Lambda
ETL için tasarım	✓ Evet	✖ Hayır
Serverless	✓	✓

Özellik	AWS Glue	Lambda
Büyük veri	✓	✗
Schema discovery	✓ Glue Crawler	✗
Operasyonel yük	Düşük	Yüksek

📌 **Sınav kuralı:**

“ETL + S3 + analytics” → **Glue**

🔥 **Neden Apache Parquet? (Çok Sorulur)**

- Columnar format
- Daha az S3 scan maliyeti
- Athena & QuickSight **çok hızlı**
- JSON/CSV'ye göre **çok daha ucuz**

📌 **Sınavda:**

“Analytics optimized format” → **Parquet**

🧠 **Lake Formation Ne Zaman Çıkar?**

Lake Formation şu kelimelerle birlikte gelir:

- “Centralized data lake”
- “Multiple data sources”
- “Least operational overhead”
- “Fine-grained access control”

👉 Bu soruda **tam işaret**.

## QUESTION 27

A company stores data in an Amazon Aurora PostgreSQL DB cluster. The company must store all the data for 5 years and must delete all the data after 5 years. The company also must indefinitely keep audit logs of actions that are performed within the database. Currently, the company has automated backups configured for Aurora.

Which combination of steps should a solutions architect take to meet these requirements? (Choose two.)

- A. Take a manual snapshot of the DB cluster.
- B. Create a lifecycle policy for the automated backups.
- C. Configure automated backup retention for 5 years.
- D. Configure an Amazon CloudWatch Logs export for the DB cluster.
- E. Use AWS Backup to take the backups and to keep the backups for 5 years.

**Soru:**

Bir şirket, verilerini **Amazon Aurora PostgreSQL DB cluster** üzerinde saklamaktadır. Şirket, **tüm verileri 5 yıl boyunca saklamak ve 5 yılın sonunda tüm verileri silmek** zorundadır. Şirket ayrıca, veritabanı içinde gerçekleştirilen **tüm işlemlere ait denetim (audit) günlüklerini süresiz (indefinitely) olarak saklamak** zorundadır. Şu anda şirket, Aurora için **otomatik yedeklemeleri (automated backups)** yapılandırmış durumdadır.

Bu gereksinimleri karşılamak için bir çözümler mimarı **hangi iki adımı** atmalıdır? (**İki seçenek seçin.**)

- A. DB cluster için **manuel bir snapshot** almak.
- B. **Otomatik yedeklemeler** için bir **lifecycle (yaşam döngüsü) politikası** oluşturmak.
- C. **Otomatik yedekleme saklama süresini 5 yıl** olarak yapılandırmak.
- D. DB cluster için **Amazon CloudWatch Logs dışa aktarma (export)** yapılandırmak.
- E. **AWS Backup** kullanarak yedekleme almak ve yedekleri **5 yıl boyunca saklamak**.

**Soru Analizi:**

**Veri saklama gereksinimi (DB verisi)**

- **Tüm veriler 5 yıl saklanmalı**
- **5 yılın sonunda mutlaka silinmeli**

📌 Bu:

- Zaman bazlı
- Yasal/regülasyon odaklı
- **Kontrollü retention** gerektirir

**Audit log gereksinimi**

- DB içindeki **tüm aksiyonların log'ları**
- **Süresiz (indefinitely)** saklanmalı

📌 Bu:

- DB verisinden **bağımsız**
- Silinmemeli
- Ayrı bir log servisine aktarılmalı

### Mevcut durum

- Aurora'da **automated backups zaten açık**

👉 Yani:

- Yeni bir backup mekanizması değil
- **Doğru retention + doğru log export** aranıyor

### Seçenek Analizi:

D. Configure an Amazon CloudWatch Logs export for the DB cluster

Neden doğru?

- Aurora PostgreSQL:
  - Audit, error, general log'ları
  - **CloudWatch Logs'a export edebilir**
- CloudWatch Logs:
  - **Süresiz saklanabilir**
  - Retention isteğe bağlı

📌 Audit log'lar DB'den **bağımsız korunur**

E. Use AWS Backup to take the backups and to keep the backups for 5 years

Neden doğru?

- **AWS Backup:**
  - Centralized backup
  - **Yıllarca retention destekler**
  - Otomatik silme (expiry) sağlar
- 5 yıl sonunda:
  - Backup'lar otomatik silinir

- “must delete after 5 years” şartı sağlanır

📌 **Regülasyonlu retention için doğru servis**

✗ **A. Take a manual snapshot of the DB cluster**

✗ **Neden yanlış?**

- Manual snapshot:

- **Süresiz tutulur**
- Otomatik silinmez

- Soruda:

“must delete all the data after 5 years”

📌 **Manual snapshot → retention kontrolü yok**

✗ **B. Create a lifecycle policy for the automated backups**

✗ **Neden yanlış?**

- Aurora automated backups:

- **S3 lifecycle policy desteklemez**
- Retention ayarı **DB seviyesinde yapılır**

📌 **Teknik olarak geçersiz seçenek**

✗ **C. Configure automated backup retention for 5 years**

✗ **Neden yanlış?**

- Aurora automated backups:

- **Maksimum 35 gün retention**
- 5 yıl = **mümkün değil**

📌 **AWS sınavında:**

“Aurora automated backup retention ≤ 35 days”

🎯 **Sonuç**

“**Aurora uzun süreli backup = AWS Backup**

**Audit log = CloudWatch Logs”**

🧠 **Ek Sınav İpuçları (Çok Çıkar)**

- ✗ Automated backups ≠ long-term retention
- ✗ Manual snapshot ≠ controlled deletion
- ✓ AWS Backup = compliance + expiry
- ✓ CloudWatch Logs = audit logs

### 🔥 Aurora Backups – Snapshot – AWS Backup (Net Karşılaştırma)

Özellik	Automated Backups	Manual Snapshot	AWS Backup
Retention süresi	Max 35 gün	Sınırsız	Yıllar (policy ile)
Otomatik silme	✗	✗	✓
Compliance / regülasyon	✗	✗	✓
Centralized yönetim	✗	✗	✓
Sınavda anlamı	Kısa süreli DR	Anlık kopya	Uzun süreli saklama

### 📌 Altın kural:

“Yıl bazlı retention” → **AWS Backup**

### 🔥 Aurora Audit Logs – Nerede, Nasıl?

Aurora PostgreSQL'de audit log'lar:

- error
- general
- slow query
- audit (pgAudit ile)

**Doğru yaklaşım:**

- **CloudWatch Logs export**
- Retention: **Never expire** (indefinite)

### 📌 Sınav ezberi:

“Audit logs must be kept indefinitely” → **CloudWatch Logs**

## Sınav Tuzakları (Mutlaka Bil)

-  “Automated backups 5 yıl” → **İmkânsız**
  -  “Manual snapshot + delete later” → **Kontrol yok**
  -  “Lifecycle policy on RDS backups” → **Yok**
  -  “Controlled delete after X years” → **AWS Backup**
- 

## QUESTION 28

A solutions architect is optimizing a website for an upcoming musical event. Videos of the performances will be streamed in real time and then will be available on demand. The event is expected to attract a global online audience.

Which service will improve the performance of both the real-time and on-demand streaming?

- A. Amazon CloudFront
- B. AWS Global Accelerator
- C. Amazon Route 53
- D. Amazon S3 Transfer Acceleration

### Soru:

Bir çözümler mimarı, yaklaşan bir **müzik etkinliği** için bir web sitesini optimize etmektedir. Performans videoları **gerçek zamanlı (real-time) olarak yayınlanacak** ve ardından **isteğe bağlı (on-demand)** olarak erişilebilir olacaktır. Etkinliğin **küresel (global) bir çevrim içi izleyici kitlesi** çekmesi beklenmektedir.

**Gerçek zamanlı yayın ve isteğe bağlı yayın** performansını birlikte artıracak **hangi servis** bu gereksinimleri karşılar?

- A. Amazon CloudFront
- B. AWS Global Accelerator
- C. Amazon Route 53
- D. Amazon S3 Transfer Acceleration

### Soru Analizi:

#### İçerik türü

-  **Video streaming**
- Hem:

- **Real-time (canlı yayın)**
- **On-demand (sonradan izleme)**

👉 Yani **media delivery** senaryosu.

### Kullanıcı kitlesi

-  **Global audience**

👉 Coğrafi olarak dağıtık kullanıcılar → **edge location** ihtiyacı.

### Amaç

- **Performance optimization**
- Hem canlı hem isteğe bağlı içerik için **tek servis**

📌 Bu, klasik **CDN (Content Delivery Network)** tanımıdır.

### Seçenek Analizi:

#### A. Amazon CloudFront

##### Neden doğru?

- AWS'nin **CDN servisi**
- Edge location'lar üzerinden:
  - Düşük gecikme
  - Yüksek throughput
- Destekler:
  - **Live streaming**
  - **Video on-demand (VOD)**
- Medya dağıtıımı için özel optimize edilmiştir

📌 **Streaming + global + performance = CloudFront**

#### B. AWS Global Accelerator

##### Neden yanlış?

- Network-level routing optimizasyonu yapar
- **Cache yok**
- Media streaming için tasarlanmamıştır

✖ **Routing ≠ streaming optimization**

✖ **C. Amazon Route 53**

✖ **Neden yanlış?**

- DNS servisi
- Yalnızca **isim çözümleme**
- İçerik taşımaz, hızlandırmaz

✖ **DNS ≠ content delivery**

✖ **D. Amazon S3 Transfer Acceleration**

✖ **Neden yanlış?**

- Amaç:
  - S3'e yükleme (**upload**) hızlandırmak
- Download / streaming performansı için değil

✖ **Upload acceleration ≠ video streaming**

🎯 **Sonuç**

**“Global video streaming = CloudFront”**

🧠 **Mini Ezber Tablosu**

Servis	Ne zaman?
CloudFront	Video, CDN, streaming
Global Accelerator	TCP/UDP app
Route 53	DNS
S3 Transfer Acceleration	Global upload

---

## QUESTION 29

A company is running a publicly accessible serverless application that uses Amazon API Gateway and AWS Lambda. The application's traffic recently spiked due to fraudulent requests from botnets.

Which steps should a solutions architect take to block requests from unauthorized users? (Choose two.)

- A. Create a usage plan with an API key that is shared with genuine users only.
- B. Integrate logic within the Lambda function to ignore the requests from fraudulent IP addresses.
- C. Implement an AWS WAF rule to target malicious requests and trigger actions to filter them out.
- D. Convert the existing public API to a private API. Update the DNS records to redirect users to the new API endpoint.
- E. Create an IAM role for each user attempting to access the API. A user will assume the role when making the API call.

**Soru:**

Bir şirket, **Amazon API Gateway** ve **AWS Lambda** kullanan, **herkese açık (publicly accessible)** bir serverless uygulama çalıştırmaktadır. Uygulamanın trafiği, **botnet'lerden gelen sahte (fraudulent) istekler** nedeniyle son zamanlarda ani bir artış göstermiştir.

Yetkisiz kullanıcıların isteklerini **engellemek** için bir çözümler mimarı **hangi iki adımı** atmalıdır? (**İki seçenek seçin.**)

- A. Yalnızca gerçek kullanıcılarla paylaşılan bir **API anahtarı (API key)** içeren bir **usage plan** oluşturmak.
- B. Sahte IP adreslerinden gelen istekleri yok saymak için **Lambda fonksiyonu içinde mantık (logic)** eklemek.
- C. Kötü niyetli istekleri hedefleyen ve bunları **filtrelemek için aksiyonlar tetikleyen bir AWS WAF kuralı** uygulamak.
- D. Mevcut herkese açık API'yi **private API**'ye dönüştürmek ve kullanıcıları yeni API uç noktasına yönlendirmek için **DNS kayıtlarını güncellemek**.
- E. API'ye erişmeye çalışan her kullanıcı için bir **IAM rolü** oluşturmak ve API çağrıları sırasında kullanıcının bu rolü üstlenmesini sağlamak.

**Soru Analizi:**

Soruda geçen **anahtar ifadeler**:

**Mimari**

- **Serverless**
- **Amazon API Gateway + AWS Lambda**
- **Publicly accessible** (herkese açık)

👉 Yani internetten doğrudan erişilebilen bir API.

## Problem

- **Traffic spike**
- **Botnets**
- **Fraudulent (sahte) requests**

👉 Bu:

- DDoS benzeri
- Otomatik bot trafiği
- Yetkisiz erişim problemi

## Amaç

### Block requests from unauthorized users

📌 Önemli:

- Sadece loglamak değil
- Lambda içinde filtrelemek değil
- **En erken noktada engellemek**

📌 Ayrıca AWS sınavında:

**LEAST cost + BEST PRACTICE** örtük olarak aranır.

## Seçenek Analizi:

### A. Usage plan + API key

*Create a usage plan with an API key...*

### Neden doğru?

- API Gateway:
  - API key gerektirecek şekilde yapılandırılabilir
- Botlar:
  - API key olmadan çağrı yapamaz
- Ek avantaj:
  - **Rate limiting**
  - **Throttling**

- ✖️ **Unauthorized users** → API key ile engellenir
- ✖️ Lambda'ya gitmeden engeller → **ucuz + etkili**

### ✓ C. AWS WAF kuralı

*Implement an AWS WAF rule...*

### ✓ Neden doğru?

- AWS WAF:
  - Bot
  - IP
  - Rate-based
  - Managed rule sets
- API Gateway ile **native entegrasyon**
- Trafiği **en önde** keser

### ✖️ Botnets + public API = WAF

### ✗ B. Lambda içinde IP kontrolü

*Integrate logic within the Lambda function...*

### ✗ Neden yanlış?

- İstek:
  - Zaten Lambda'ya ulaşmış oluyor
- Lambda çalışır → **maliyet oluşur**
- IP listeleri:
  - Sürekli değişir
  - Yönetimi zor

### ✖️ AWS sınavında:

“Don’t filter inside Lambda if you can filter before it”

### ✗ D. Public API → Private API

*Convert the existing public API to a private API...*

### ✗ Neden yanlış?

- Private API:

- Bu:
  - Sadece VPC içinden erişilir
- Bu:
  - Tüm mimariyi değiştirir
  - Global kullanıcıları **kırar**

 **Overkill + yanlış çözüm**

 **E. IAM role per user**

*Create an IAM role for each user...*

 **Neden yanlış?**

- Son kullanıcılar için:
  - IAM role kullanımı pratik değil
- Ölçeklenmez
- API Gateway best practice değil

 **IAM role → AWS servisleri / internal Access**

 **Sonuç**

**“Public API + bot traffic = WAF + API key”**

 **Ek Sınav İpuçları (Çok Çıkar)**

-  Lambda içi güvenlik → pahalı
-  IAM role → kullanıcı auth için değil
-  WAF → bot & attack protection
-  API key → unauthorized filtering + throttling

---

## QUESTION 30

An ecommerce company hosts its analytics application in the AWS Cloud. The application generates about 300 MB of data each month. The data is stored in JSON format. The company is evaluating a disaster recovery solution to back up the data. The data must be accessible in milliseconds if it is needed, and the data must be kept for 30 days.

Which solution meets these requirements MOST cost-effectively?

- A. Amazon OpenSearch Service (Amazon Elasticsearch Service)
- B. Amazon S3 Glacier
- C. Amazon S3 Standard
- D. Amazon RDS for PostgreSQL

**Soru:**

Bir e-ticaret şirketi, analiz (analytics) uygulamasını **AWS Cloud** üzerinde barındırmaktadır.

Uygulama, her ay yaklaşık **300 MB** veri üretmektedir.

Veriler **JSON formatında** saklanmaktadır.

Şirket, verileri **yedeklemek (backup)** için bir **felaket kurtarma (disaster recovery)** çözümünü değerlendirmektedir.

Verilere ihtiyaç duyulduğunda **milisaniyeler içinde erişilebilir** olması gerekmektedir ve veriler **30 gün boyunca saklanmalıdır**.

Bu gereksinimleri **EN maliyet etkin** şekilde karşılayan çözüm hangisidir?

- A. Amazon OpenSearch Service (Amazon Elasticsearch Service)
- B. Amazon S3 Glacier
- C. Amazon S3 Standard
- D. Amazon RDS for PostgreSQL

**Soru Analizi:**

Sorudaki anahtar ifadeleri ayıralım:

**Veri miktarı ve türü**

- **Aylık 300 MB**
- **JSON format**
- Küçük–orta ölçekli veri

👉 Büyük veri platformlarına gerek yok.

**Kullanım amacı**

- **Disaster recovery / backup**
- Aktif sorgulama veya analiz yok

👉 Yani:

- Sadece saklama
- Gerektiğinde geri okuma

## Erişim gereksinimi

“Accessible in milliseconds”

📌 Bu çok kritik:

- ✗ Glacier elenir (dakika–saat)
- ✓ S3 Standard uygun

## Saklama süresi

- **30 gün**

👉 Kısa süreli retention

👉 Archive veya DB çözümü gereksiz

## En önemli ifade

### MOST cost-effective

📌 Yani:

- Gereksiz servis yok
- Yönetim yükü yok
- Basit, ucuz, uygun performans

## Seçenek Analizi:

✓ C. Amazon S3 Standard

✓ Neden doğru?

- Milisaniyeler içinde erişim
- Küçük veri için çok ucuz
- JSON formatı için ideal
- DR / backup için yaygın kullanım
- 30 gün için **minimum maliyet + maksimum basitlik**

📌 Gerekirse:

- Lifecycle policy ile otomatik silme

✗ A. Amazon OpenSearch Service

✗ Neden yanlış?

- Log search / near real-time analytics için
- Cluster yönetimi gereklidir
- Backup için **aşırı pahalı ve karmaşık**

✖ **Analytics/search ≠ backup**

✖ **B. Amazon S3 Glacier**

✖ **Neden yanlış?**

- Çok ucuz ama:
  - **Retrieval süresi: dakika-saat**

“milliseconds”

✖ **Hızlı erişim = Glacier ASLA**

✖ **D. Amazon RDS for PostgreSQL**

✖ **Neden yanlış?**

- Managed database:
  - Sürekli çalışan instance
  - Backup için pahalı
- JSON saklamak mümkün ama:
  - DR çözümü olarak mantıksız

✖ **Database ≠ object backup**

🎯 **Sonuç**

**“Backup + milliseconds access + düşük maliyet = S3 Standard”**

🧠 **Mini Ezber Tablosu**

Senaryo	Doğru Servis
Hızlı erişimli backup	S3 Standard
Arşiv (saatler)	S3 Glacier
Arama / log	OpenSearch

Senaryo	Doğru Servis
Uygulama DB	RDS

---

### QUESTION 31

A company has a small Python application that processes JSON documents and outputs the results to an on-premises SQL database. The application runs thousands of times each day. The company wants to move the application to the AWS Cloud. The company needs a highly available solution that maximizes scalability and minimizes operational overhead.

Which solution will meet these requirements?

- A. Place the JSON documents in an Amazon S3 bucket. Run the Python code on multiple Amazon EC2 instances to process the documents. Store the results in an Amazon Aurora DB cluster.
- B. Place the JSON documents in an Amazon S3 bucket. Create an AWS Lambda function that runs the Python code to process the documents as they arrive in the S3 bucket. Store the results in an Amazon Aurora DB cluster.
- C. Place the JSON documents in an Amazon Elastic Block Store (Amazon EBS) volume. Use the EBS Multi-Attach feature to attach the volume to multiple Amazon EC2 instances. Run the Python code on the EC2 instances to process the documents. Store the results on an Amazon RDS DB instance.
- D. Place the JSON documents in an Amazon Simple Queue Service (Amazon SQS) queue as messages. Deploy the Python code as a container on an Amazon Elastic Container Service (Amazon ECS) cluster that is configured with the Amazon EC2 launch type. Use the container to process the SQS messages. Store the results on an Amazon RDS DB instance.

#### Soru:

Bir şirketin, **JSON belgelerini işleyen** ve sonuçları **şirket içi (on-premises)** bir **SQL veritabanına** yazan **küçük bir Python uygulaması** vardır. Uygulama **günde binlerce kez** çalışmaktadır. Şirket, uygulamayı **AWS Cloud** ortamına taşımak istemektedir. Şirket, **yüksek erişilebilirliğe sahip, ölçeklenebilirliği en üst düzeye çıkaran ve operasyonel yükü en aza indiren** bir çözüme ihtiyaç duymaktadır.

Bu gereksinimleri **hangi çözüm** karşılar?

**A.** JSON belgelerini bir **Amazon S3 bucket** içine koymak. Belgeleri işlemek için Python kodunu **birden fazla Amazon EC2 instance** üzerinde çalışırmak. Sonuçları bir **Amazon Aurora DB cluster** içinde saklamak.

**B.** JSON belgelerini bir **Amazon S3 bucket** içine koymak. Belgeler S3 bucket'a ulaştığında Python kodunu çalışıran bir **AWS Lambda fonksiyonu** oluşturmak. Sonuçları bir **Amazon Aurora DB cluster** içinde saklamak.

**C.** JSON belgelerini bir **Amazon Elastic Block Store (Amazon EBS)** volume içine koymak. **EBS Multi-Attach** özelliğini kullanarak bu volume'u birden fazla **Amazon EC2 instance**'a bağlamak. Python kodunu EC2 instance'lar üzerinde çalıştırarak belgeleri işlemek. Sonuçları bir **Amazon RDS DB instance** içinde saklamak.

**D.** JSON belgelerini mesaj olarak bir **Amazon Simple Queue Service (Amazon SQS)** kuyruğuna koymak. Python kodunu bir konteyner olarak **Amazon Elastic Container Service (Amazon ECS)** kümesi üzerinde (**Amazon EC2 launch type** ile) çalışırmak. Konteyneri kullanarak SQS mesajlarını işlemek. Sonuçları bir **Amazon RDS DB instance** içinde saklamak.

#### Soru Analizi:

Sorudaki kritik ifadeler:

##### Uygulama tipi

- **Küçük Python uygulaması**
- **JSON belgeleri işliyor**
- **Günde binlerce kez çalışıyor**

👉 Kısa süreli, event-driven işler için uygun.

##### Hedef mimari

- **AWS Cloud**
- **High availability**
- **Maximize scalability**
- **Minimize operational overhead**

📌 Bu üçlü AWS sınavlarında genelde:

**Serverless** çözümü işaret eder.

##### Çıktı

- Sonuçlar bir **SQL veritabanına** yazılıyor

- Cloud tarafından:
  - Managed
  - Highly available

### **Seçenek Analizi:**

#### **B. S3 + AWS Lambda + Aurora**

##### **Neden doğru?**

- **S3 event → Lambda**
- Lambda:
  - Otomatik ölçeklenir
  - Sunucu yok
  - Yük kadar ödeme
- Aurora:
  - Managed
  - Highly available

#### **En düşük operasyonel yük + en yüksek ölçeklenebilirlik**

**B.** JSON belgelerini bir **Amazon S3 bucket** içine koymak, belgeler S3 bucket'a ulaştığında Python kodunu çalıştırın bir **AWS Lambda fonksiyonu** oluşturmak ve sonuçları bir **Amazon Aurora DB cluster** içinde saklamak.

#### Bu seçenek:

- **Yüksek erişilebilirlik** sağlar
- **Otomatik ölçeklenir** (günde binlerce çalışma için ideal)
- **Sunucu yönetimi gerektirmez** (en düşük operasyonel yük)
- **Serverless mimari** olduğu için sınavda beklenen çözümdür

#### **A. S3 + EC2 + Aurora**

##### **Neden uygun değil?**

- EC2:
  - Sunucu yönetimi
  - Patch, scaling, monitoring gereklidir

- Operational overhead **yüksek**

📌 Çalışır ama **en iyi çözüm değil**

## ✗ C. EBS Multi-Attach + EC2 + RDS

### Neden yanlış?

- EBS:
  - Paylaşımlı dosya sistemi değil
- Multi-Attach:
  - Çok sınırlı kullanım senaryosu
- EC2 yönetimi:
  - Yüksek operasyonel yük

📌 Mimari olarak hatalıya yakın

## ✗ D. SQS + ECS (EC2 launch type) + RDS

### Neden elendi?

- ECS EC2 launch type:
  - EC2 yönetimi gereklidir
- Container orchestration:
  - Lambda'ya göre daha karmaşık
- Overhead daha yüksek

📌 İyi ama **en az operasyonel yük değil**

## ⌚ Sonuç

“Event-driven, sık çalışan küçük işler → Lambda”

## 🧠 Karşılaştırma Özeti

Kriter                  En iyi

Ölçeklenebilirlik Lambda

Operasyonel yük Lambda

HA                  Aurora + Lambda

## 📝 EC2'Yİ ELEYEN ANAHTAR KELİMELER

🚫 1 “Minimize operational overhead”

➡ EC2 = sunucu yönetimi

➡ Elenir, yerine:

- Lambda
- Fargate
- Managed services

🚫 2 “Serverless”

Direkt işaret fişi 🚨

➡ EC2 kesin elenir

🚫 3 “Automatically scales with demand”

EC2:

- Auto Scaling ister
- Manuel ayar gereklidir

➡ Lambda / DynamoDB / S3

🚫 4 “Event-driven”

“when files arrive”

“on upload”

“on request”

➡ Lambda

➡ EC2 elenir

🚫 5 “Runs thousands of times per day”

Kısa, sık tetiklenen işler:

➡ Lambda

➡ EC2 gereksiz

🚫 6 “Pay only when the code runs”

➡ Lambda

➡ EC2 her zaman ücretlidir

🚫 7 “No servers to manage”

➡ EC2 = server

  “Most cost-effective” (küçük iş yükü)

EC2:

- Instance maliyeti
- Idle cost

 **Serverless daha ucuz**

  “Highly available by default”

EC2:

- HA manuel kurulur

 **Managed / serverless**

  “Simple / small application”

EC2:

- Overkill

 **Lambda**

 **SINAV TUZAĞI KELİMELERİ**

Bu kelimeler **EC2’yi cazip gösterir ama tuzaktır**:

- “multiple EC2 instances”
- “Auto Scaling group”
- “load balancer”

 Eğer **serverless alternatif** varsa → **EC2 yine elenir**

---

### QUESTION 32

A company wants to use high performance computing (HPC) infrastructure on AWS for financial risk modeling. The company’s HPC workloads run on Linux. Each HPC workflow runs on hundreds of Amazon EC2 Spot Instances, is short-lived, and generates thousands of output files that are ultimately stored in persistent storage for analytics and long-term future use. The company seeks a cloud storage solution that permits the copying of on-premises data to long-term persistent storage to make data available for processing by all EC2 instances. The solution should also be a high performance file system that is integrated with persistent storage to read and write datasets and output files.

Which combination of AWS services meets these requirements?

- A. Amazon FSx for Lustre integrated with Amazon S3
- B. Amazon FSx for Windows File Server integrated with Amazon S3
- C. Amazon S3 Glacier integrated with Amazon Elastic Block Store (Amazon EBS)
- D. Amazon S3 bucket with a VPC endpoint integrated with an Amazon Elastic Block Store (Amazon EBS) General Purpose SSD (gp2) volume

**Soru:**

Bir şirket, **finansal risk modellemesi** için AWS üzerinde **yüksek performanslı bilgi işlem (HPC)** altyapısı kullanmak istemektedir. Şirketin HPC iş yükleri **Linux** üzerinde çalışmaktadır. Her bir HPC iş akışı, **yüzlerce Amazon EC2 Spot Instance** üzerinde çalışır, **kısa sürelidir ve binlerce çıktı dosyası** üretir. Bu çıktı dosyaları, en sonunda **analitik ve uzun vadeli gelecekteki kullanım** için **kalıcı (persistent) depolama** alanında saklanır.

Şirket, **on-premises verilerin uzun vadeli kalıcı depolamaya kopyalanmasına** izin veren ve bu verilerin **tüm EC2 instance'ları tarafından işlenebilir olmasını** sağlayan bir **bulut depolama çözümü** aramaktadır. Çözüm ayrıca, veri setlerini ve çıktı dosyalarını **okumak ve yazmak** için **kalıcı depolama ile entegre, yüksek performanslı bir dosya sistemi** olmalıdır.

Bu gereksinimleri **hangi AWS servis kombinasyonu** karşılar?

- A. Amazon FSx for Lustre ile Amazon S3 entegrasyonu
- B. Amazon FSx for Windows File Server ile Amazon S3 entegrasyonu
- C. Amazon S3 Glacier ile Amazon Elastic Block Store (Amazon EBS) entegrasyonu
- D. Bir Amazon S3 bucket'ının, bir VPC endpoint aracılığıyla, Amazon Elastic Block Store (Amazon EBS) General Purpose SSD (gp2) volume ile entegre edilmesi

**Soru Analizi:**

Sorudaki kritik ifadeler:

**İş yükü tipi**

- **High Performance Computing (HPC)**
- **Finansal risk modelleme**
- **Linux**

📌 HPC → çok düşük gecikme, çok yüksek throughput

📌 Linux → Windows FS elenir

## Çalışma şekli

- Yüzlerce EC2 Spot Instance
- Kısa süreli (short-lived)
- Binlerce çıktı dosyası

📌 Tüm instance'ların **aynı dosya sistemine** eşzamanlı erişmesi gereklidir  
📌 **Paylaşımı, paralel dosya sistemi** şart

## Depolama gereksinimi

- Persistent storage
- Uzun vadeli saklama
- Analytics için erişim

📌 HPC çalışırken → **yüksek performans**  
📌 İş bittikten sonra → **ucuz, kalıcı depolama**

## En kritik ifade

**High performance file system integrated with persistent storage**

📌 Bu ifade AWS'de **tek bir servisi işaret eder**:

**FSx for Lustre + S3**

**Seçenek Analizi:**

A. Amazon FSx for Lustre + Amazon S3

**Neden doğru?**

- **Lustre:**
  - HPC için özel tasarlanmış
  - Paralel dosya sistemi
  - Çok yüksek IOPS / throughput
- **S3 entegrasyonu:**
  - Veriler S3'te kalıcı olarak saklanır
  - FSx → geçici ama ultra hızlı
- On-prem → S3 → FSx:
  - Soruda istenen mimari birebir

⭐ AWS'de HPC = FSx for Lustre

✗ B. FSx for Windows File Server + S3

**Neden yanlış?**

- Windows tabanlı
- HPC ve Linux için uygun değil
- SMB protokolü → yüksek gecikme

✗ C. S3 Glacier + EBS

**Neden yanlış?**

- Glacier:
  - Arşiv
  - Saatler süren erişim
- HPC için **tamamen uygunsuz**

✗ D. S3 + EBS (gp2)

**Neden yanlış?**

- EBS:
  - Tek instance'a bağlı
  - Paylaşımlı dosya sistemi değil
- HPC yüzlerce instance → çalışmaz

⌚ Sonuç

“HPC + Linux + shared file system + S3 = FSx for Lustre”

🕒 Hızlı Karşılaştırma

**Servis      HPC Uygunluğu**

FSx for Lustre 

FSx Windows 

EBS 

S3 Glacier 

---

### QUESTION 33

A company is building a containerized application on premises and decides to move the application to AWS. The application will have thousands of users soon after it is deployed. The company is unsure how to manage the deployment of containers at scale. The company needs to deploy the containerized application in a highly available architecture that minimizes operational overhead.

Which solution will meet these requirements?

- A. Store container images in an Amazon Elastic Container Registry (Amazon ECR) repository. Use an Amazon Elastic Container Service (Amazon ECS) cluster with the AWS Fargate launch type to run the containers. Use target tracking to scale automatically based on demand.
- B. Store container images in an Amazon Elastic Container Registry (Amazon ECR) repository. Use an Amazon Elastic Container Service (Amazon ECS) cluster with the Amazon EC2 launch type to run the containers. Use target tracking to scale automatically based on demand.
- C. Store container images in a repository that runs on an Amazon EC2 instance. Run the containers on EC2 instances that are spread across multiple Availability Zones. Monitor the average CPU utilization in Amazon CloudWatch. Launch new EC2 instances as needed.
- D. Create an Amazon EC2 Amazon Machine Image (AMI) that contains the container image. Launch EC2 instances in an Auto Scaling group across multiple Availability Zones. Use an Amazon CloudWatch alarm to scale out EC2 instances when the average CPU utilization threshold is breached.

**Soru:**

Bir şirket, kurum içinde (on-premises) kapsayıcı (container) tabanlı bir uygulama geliştiriyor ve uygulamayı AWS'ye taşımaya karar veriyor. Uygulamanın, dağıtıldıktan kısa süre sonra binlerce kullanıcı olması bekleniyor. Şirket, kapsayıcıları ölçeklendirilmiş bir şekilde nasıl yöneteceğini emin değil. Şirket, kapsayıcı tabanlı uygulamayı **yüksek kullanılabilirlik ile ve operasyonel yükü en aza indirerek** dağıtmak istiyor.

Hangi çözüm bu gereksinimleri karşılar?

- A. Kapsayıcı görüntülerini (container images) **Amazon Elastic Container Registry (Amazon ECR)** deposunda saklayın. Kapsayıcıları çalıştırmak için **AWS Fargate** başlatma tipi ile bir **Amazon ECS**kümesi kullanın. Talebe göre otomatik ölçekleme için **target tracking** kullanın.

B. Kapsayıcı görüntülerini **Amazon ECR** deposunda saklayın. Kapsayıcıları çalıştırmak için **Amazon EC2** başlatma tipi ile bir **Amazon ECS** kümesi kullanın. Talebe göre otomatik ölçekleme için **target tracking** kullanın.

C. Kapsayıcı görüntülerini bir **Amazon EC2** üzerinde çalışan bir depoda saklayın. Kapsayıcıları birden çok Erişilebilirlik Alanına (Availability Zone) yayılan EC2 örneklerinde çalıştırın. Ortalama CPU kullanımını **Amazon CloudWatch** ile izleyin. Gerekçinde yeni EC2 örnekleri başlatın.

D. Kapsayıcı görüntüsünü içeren bir **Amazon EC2 AMI** oluşturun. EC2 örneklerini bir **Auto Scaling grubu** içinde birden çok Erişilebilirlik Alanına başlatın. Ortalama CPU kullanım eşiği aşıldığında ölçekleme yapmak için **Amazon CloudWatch alarmı** kullanın.

#### Soru Analizi:

#### Sorunun Gereksinimleri:

1. **Kapsayıcı tabanlı uygulama** (containerized application).
2. **AWS'ye taşınacak**, yani bulut ortamında çalışacak.
3. **Binlerce kullanıcı** olacak → ölçeklenebilirlik çok önemli.
4. **Yüksek kullanılabilirlik** (high availability) gereklidir.
5. **Operasyonel yükü en aza indirmek** → yönetim kolaylığı, otomatik ölçekleme tercih ediliyor.

Anahtar ifadeler: *kapsayıcılar, yüksek kullanılabilirlik, otomatik ölçekleme, düşük operasyonel yük.*

#### Seçenek Analizi:

##### A. ECR + ECS + Fargate + target tracking

- **ECR:** Kapsayıcı imajlarını saklamak için uygun.
- **ECS + Fargate:** Sunucu yönetmeye gerek yok, AWS altyapısı kapsayıcıları otomatik olarak çalıştırır.
- **Target tracking:** Talebe göre otomatik ölçekleme sağlar.
- **Avantaj:** Minimum operasyonel yük + yüksek kullanılabilirlik + otomatik ölçekleme.

Bu seçenek, gereksinimleri tamamen karşılıyor.

##### B. ECR + ECS + EC2 + target tracking

- **ECR:** Uygun.

- **ECS + EC2:** Kapsayıcıları çalıştırmak için EC2 kullanıyor. EC2 örneklerini yönetmek gerekiyor (patch, kapasite planlama, Availability Zone dağılımı). 
- **Target tracking:** ECS ile otomatik ölçekleme mümkün, ama EC2 örnekleri için ek yönetim gerekebilir.
- **Dezavantaj:** Operasyonel yük Fargate'e göre daha fazla.

Çalışır ama yönetim yükü daha yüksek.

### C. Kendi EC2 repository + EC2 üzerinde kapsayıcılar + CloudWatch CPU

- Tüm altyapıyı kendiniz yönetiyorsunuz: repository, EC2 örnekleri, ölçekleme. 
- CloudWatch CPU kullanımı ile ölçekleme basit bir mekanizma. Ancak **yüksek kullanılabilirlik ve otomatik ölçekleme için elle yönetim gereklidir.**
- Operasyonel yük çok yüksek. 

Gereksinimleri karşılamıyor.

### D. EC2 AMI + Auto Scaling + CloudWatch alarm

- EC2 AMI ile her örnek kapsayıcı içeriyor. 
- Auto Scaling kullanıyor → bazı otomatik ölçekleme imkanı.
- Ama EC2 örneklerini kendiniz yönetiyorsunuz (yüksek operasyonel yük). 
- CloudWatch CPU alarmına dayalı ölçekleme, gerçek container talebini yansıtmayabilir. 

Yüksek kullanılabilirlik mümkün ama operasyonel yük fazla ve ölçekleme container odaklı değil.

### Sonuç

#### En uygun çözüm: A. ECR + ECS + Fargate + target tracking

- Minimum operasyonel yük → sunucu yönetimi yok
- Yüksek kullanılabilirlik → AWS çoklu Availability Zone yönetiyor
- Otomatik ölçekleme → target tracking ile talebe göre ölçekleniyor
- Container odaklı → modern ve yönetimi kolay

### QUESTION 34

A company has two applications: a sender application that sends messages with payloads to be processed and a processing application intended to receive the messages with payloads. The company wants to implement an AWS service to handle messages between the two applications. The sender application can send about 1,000 messages each hour. The messages may take up to 2 days to be processed: If the messages fail to process, they must be retained so that they do not impact the processing of any remaining messages.

Which solution meets these requirements and is the MOST operationally efficient?

- A. Set up an Amazon EC2 instance running a Redis database. Configure both applications to use the instance. Store, process, and delete the messages, respectively.
- B. Use an Amazon Kinesis data stream to receive the messages from the sender application. Integrate the processing application with the Kinesis Client Library (KCL).
- C. Integrate the sender and processor applications with an Amazon Simple Queue Service (Amazon SQS) queue. Configure a dead-letter queue to collect the messages that failed to process.
- D. Subscribe the processing application to an Amazon Simple Notification Service (Amazon SNS) topic to receive notifications to process. Integrate the sender application to write to the SNS topic.

**Soru:**

Bir şirketin iki uygulaması vardır: bir mesaj gönderen uygulama (sender application) ve mesajları almak üzere tasarlanmış bir işleme uygulaması (processing application). Şirket, iki uygulama arasında mesajları yönetmek için bir AWS hizmeti uygulamak istiyor. Gönderici uygulama saatte yaklaşık 1.000 mesaj gönderebiliyor. Mesajların işlenmesi 2 güne kadar sürebilir. Eğer mesajlar işlenemezse, bu mesajlar saklanmalı ve kalan mesajların işlenmesini etkilememelidir.

Hangi çözüm bu gereksinimleri karşılar ve en **operasyonel olarak verimli** olur?

- A. Bir Amazon EC2 örneğinde Redis veritabanı kurun. Her iki uygulamayı da bu örneği kullanacak şekilde yapılandırın. Mesajları sırasıyla depolayın, işleyin ve silin.
- B. Gönderici uygulamadan mesajları almak için bir Amazon Kinesis veri akışı kullanın. İşleme uygulamasını Kinesis Client Library (KCL) ile entegre edin.
- C. Gönderici ve işleme uygulamasını bir Amazon Simple Queue Service (Amazon SQS) kuyruğu ile entegre edin. İşlenemeyen mesajları toplamak için bir **dead-letter queue** yapılandırın.

D. İşleme uygulamasını bir Amazon Simple Notification Service (Amazon SNS) konusu ile abone yapın ve mesajları işlemek için bildirim alın. Gönderici uygulamayı SNS konusuna yazacak şekilde entegre edin.

#### Soru Analizi:

##### Sorunun Gereksinimleri

- Mesaj tabanlı iletişim:** Gönderici ve işleyici uygulamalar arasında mesajlar iletilecek.
- Mesaj hacmi:** Saatte ~1.000 mesaj → orta ölçekli bir yük.
- Mesajların saklanması:** İşlenme başarısız olursa mesajlar kaybolmamalı, diğer mesajların işlenmesini engellememeli.
- Mesajların işlenme süresi:** 2 güne kadar sürebilir → mesajlar uzun süre saklanabilmeli.
- Operasyonel verimlilik:** Yönetimi minimum, sunucu kurup yönetmekten kaçınmak tercih ediliyor.

Anahtar ifadeler: *mesaj kuyruğu, dead-letter queue, uzun süreli saklama, operasyonel kolaylık.*

#### Seçenek Analizi:

##### C. SQS + Dead-Letter Queue

- SQS**, mesaj kuyruğu hizmetidir, mesajları güvenli şekilde saklar ve işlenmeyi garanti eder.
- Dead-letter queue (DLQ)** ile işlenemeyen mesajlar ayrı bir kuyruğa atılır → diğer mesajlar etkilenmez.
- Mesajlar **maksimum 14 gün** saklanabilir → 2 gün için yeterli.
- Operasyonel yük minimum:** yönetilen servis, sunucu gerekmmez.
- Mesajların güvenli ve uzun süre saklanması için ideal çözüm.

##### A. EC2 + Redis

- Redis, hızlı ve bellek tabanlıdır; kalıcı depolama için ek yapılandırma gereklidir.
- EC2'yi ve Redis'i yönetmek gereklidir (patch, yedekleme, ölçekleme).
- Mesajlar uzun süreli saklanacaksa Redis tek başına uygun değildir.
- Operasyonel yük yüksek.**

## B. Kinesis Data Stream

- Kinesis, yüksek hacimli veri akışları için uygundur.
- İşleme uygulamasını KCL ile entegre etmek gereklidir → ek geliştirme ve yönetim gereklidir.
- Mesajlar en fazla **7 gün** saklanabilir (default retention 24 saat, max 7 gün).
- Bu senaryoda mesajlar **2 gün boyunca saklanmalı**, Kinesis bunu karşılayabilir ama orta ölçekli mesajlar için biraz fazla karmaşık olabilir. 

## D. SNS

- SNS, yayın/abone (publish/subscribe) modelidir.
- Mesajlar sadece **abonelere iletilir**, kuyruk gibi saklanmaz.
- İşleme başarısız olursa mesaj kaybolabilir → gereksinimi karşılamaz. 

## Sonuç

En uygun çözüm: C. SQS + Dead-Letter Queue 

- Mesajlar güvenli şekilde saklanır
- İşlenemeyen mesajlar diğerlerini etkilemez
- Operasyonel yük minimum
- Mesajlar 2 gün boyunca saklanabilir

---

## QUESTION 35

A solutions architect must design a solution that uses Amazon CloudFront with an Amazon S3 origin to store a static website. The company's security policy requires that all website traffic be inspected by AWS WAF.

How should the solutions architect comply with these requirements?

- Configure an S3 bucket policy to accept requests coming from the AWS WAF Amazon Resource Name (ARN) only.
- Configure Amazon CloudFront to forward all incoming requests to AWS WAF before requesting content from the S3 origin.
- Configure a security group that allows Amazon CloudFront IP addresses to access Amazon S3 only. Associate AWS WAF to CloudFront.

D. Configure Amazon CloudFront and Amazon S3 to use an origin access identity (OAI) to restrict access to the S3 bucket. Enable AWS WAF on the distribution.

#### Soru:

Bir çözüm mimarı (solutions architect), **Amazon CloudFront** ile bir **Amazon S3** kaynağını (origin) kullanan bir statik web sitesi çözümü tasarlamak zorundadır. Şirketin güvenlik politikası, **tüm web sitesi trafiğinin AWS WAF tarafından denetlenmesini** zorunlu kılmaktadır.

Çözüm mimarı, bu gereksinimleri nasıl karşılamalıdır?

- A. S3 bucket politikasını, yalnızca **AWS WAF Amazon Resource Name (ARN)** üzerinden gelen isteklere izin verecek şekilde yapılandırın.
- B. Amazon CloudFront'u, tüm gelen istekleri **S3 kaynağından içerik talep etmeden önce AWS WAF'a iletecek** şekilde yapılandırın.
- C. Amazon CloudFront IP adreslerinin Amazon S3'e erişmesine izin veren bir **security group** yapılandırın. AWS WAF'ı CloudFront'a bağlayın.
- D. Amazon CloudFront ve Amazon S3'ü, **S3 bucket erişimini kısıtlamak için Origin Access Identity (OAI)** kullanacak şekilde yapılandırın. AWS WAF'ı dağıtıma (distribution) etkinleştirin.

#### Soru Analizi:

##### Sorunun Gereksinimleri

1. **Statik web sitesi:** Amazon S3 üzerinde barındırılıyor.
2. **CDN kullanımı:** CloudFront, S3'ü origin olarak kullanacak.
3. **Güvenlik politikası:** Tüm web sitesi trafiği **AWS WAF** ile denetlenmeli.
4. **Erişim kontrolü:** S3 doğrudan internetten erişilmemeli, sadece CloudFront üzerinden erişim olmalı.

Anahtar ifadeler: *CloudFront + S3 origin + AWS WAF + güvenli erişim.*

#### Seçenek Analizi:

- D. **CloudFront + S3 OAI kullanmak ve WAF'ı dağıtıma eklemek**
  - **OAI (Origin Access Identity):** S3 bucket'ına sadece CloudFront erişimini sağlar → güvenli erişim.
  - **AWS WAF:** CloudFront distribution'a eklenir → tüm trafik denetlenir.
  - **Avantajlar:** Operasyonel olarak yönetilen, güvenli ve AWS en iyi uygulamalarına uygun.

### A. S3 bucket politikasını AWS WAF ARN ile kısıtlamak

- **Sorun:** AWS WAF, bir kaynak değil, bir hizmettir. WAF'ın ARN'si üzerinden S3 erişimini kısıtlamak mümkün değildir. 
- **Sonuç:** Bu seçenek çalışmaz.

### B. CloudFront'u, tüm istekleri S3'e gitmeden önce WAF'a iletecek şekilde yapılandırmak

- CloudFront ile AWS WAF entegrasyonu zaten **distribution seviyesinde** sağlanır.
- Ancak "S3'e gitmeden önce WAF'a iletme" ifadesi **yanlış bir yapılandırmadır**; CloudFront dağıtımına WAF eklemek yeterlidir. 

### C. Security group ile CloudFront IP erişimi + WAF CloudFront'a eklemek

- **Sorun:** S3, bir VPC kaynağı değil, public endpoint üzerinden erişilir. Security group S3'e uygulanamaz. 
- WAF'ı CloudFront'a eklemek doğru, ama security group kısmı yanlış.

### Sonuç

#### Doğru çözüm: D. CloudFront ve S3'ü OAI ile yapılandır, WAF'ı dağıtıma etkinleştir

- S3 sadece CloudFront üzerinden erişilebilir
- Tüm trafik AWS WAF ile denetlenir
- Basit ve yönetilen bir çözüm

---

## QUESTION 36

Organizers for a global event want to put daily reports online as static HTML pages. The pages are expected to generate millions of views from users around the world. The files are stored in an Amazon S3 bucket. A solutions architect has been asked to design an efficient and effective solution.

Which action should the solutions architect take to accomplish this?

- A. Generate presigned URLs for the files.
- B. Use cross-Region replication to all Regions.
- C. Use the geoproximity feature of Amazon Route 53.
- D. Use Amazon CloudFront with the S3 bucket as its origin.

**Soru:**

Global bir etkinlik için organizatörler, **günlük raporları statik HTML sayfaları** olarak çevrimiçi yayınlamak istiyor. Sayfaların dünya genelinden milyonlarca görüntüleme olması bekleniyor. Dosyalar **Amazon S3** bucket'ında saklanıyor.

Bir çözüm mimarına, bu durumu **verimli ve etkili bir şekilde çözmesi** için görev verildi.

Hangi adımı atmalıdır?

- A. Dosyalar için **önceDEN imzALANMış URL'ler (presigned URLs)** oluşturun.
- B. Tüm bölgelerde **cross-Region replication** kullanın.
- C. Amazon Route 53'ün **geoproximity** özelliğini kullanın.
- D. **Amazon CloudFront'u** kullanarak S3 bucket'ını origin olarak ayarlayın.

**Soru Analizi:**

#### **Sorunun Gereksinimleri**

1. **Statik HTML sayfaları:** Günlük raporlar statik dosyalar olarak S3'te saklanıyor.
2. **Yüksek trafik:** Dünya genelinden milyonlarca görüntüleme bekleniyor → ölçeklenebilirlik ve düşük gecikme kritik.
3. **Verimli ve etkili çözüm:** Minimum operasyonel yönetim ve hızlı içerik dağıtıımı isteniyor.

Anahtar ifadeler: *statik web sitesi, dünya genelinde milyonlarca görüntüleme, S3, yüksek performans*.

**Seçenek Analizi:**

**D. CloudFront ile S3 origin**

- CloudFront, **küresel CDN** olarak içerikleri önbelleğe alır → yüksek trafik ve düşük gecikme sağlar.
- S3, origin olarak kalır → minimum yönetim yükü.
- En verimli ve etkili yöntem.

**A. Önceden imzalanmış URL'ler (Presigned URLs)**

- **Amaç:** Geçici erişim kontrolü sağlamak.
- **Sorun:** Bu yöntem milyonlarca kullanıcı için ölçeklendirme sağlamaz ve global erişimi optimize etmez.

**B. Cross-Region replication**

- S3 dosyalarını diğer bölgelere çoğaltır.

- Ancak **her bölgeye manuel replication** maliyetli ve karmaşıktır, ve dağıtım hızı CloudFront kadar optimize edilmez. 

### C. Route 53 geoproximity

- Trafiği coğrafi konuma göre yönlendirme sağlar. 
- Ancak statik içerik dağıtımını ve önbellekleme yapmaz → milyonlarca istek için tek başına yeterli değildir. 

### Sonuç

**Doğru çözüm: D. Amazon CloudFront ile S3 bucket'ını origin olarak kullanmak** 

- Global kullanıcılar için düşük gecikme
- Milyonlarca görüntülemeyi rahatlıkla karşılar
- Operasyonel yük minimum

---

### QUESTION 37

A company runs a production application on a fleet of Amazon EC2 instances. The application reads the data from an Amazon SQS queue and processes the messages in parallel. The message volume is unpredictable and often has intermittent traffic. This application should continually process messages without any downtime.

Which solution meets these requirements MOST cost-effectively?

- A. Use Spot Instances exclusively to handle the maximum capacity required.
- B. Use Reserved Instances exclusively to handle the maximum capacity required.
- C. Use Reserved Instances for the baseline capacity and use Spot Instances to handle additional capacity.
- D. Use Reserved Instances for the baseline capacity and use On-Demand Instances to handle additional capacity.

#### Soru:

Bir şirket, bir grup **Amazon EC2 örneği** üzerinde üretim uygulaması çalıştırıyor. Uygulama, verileri **Amazon SQS kuyruğundan** okuyor ve mesajları **paralel olarak** işliyor. Mesaj hacmi **öngörülemez** ve çoğunlukla **aralıklı trafik** gösteriyor. Bu uygulama, mesajları **kesintisiz olarak** işlemeye devam etmelidir.

Hangi çözüm, bu gereksinimleri en **maliyet etkin** şekilde karşılar?

- A. Sadece **Spot Instances** kullanarak, gereken maksimum kapasiteyi karşılayın.

- B. Sadece **Reserved Instances** kullanarak, gereken maksimum kapasiteyi karşılayın.
- C. **Reserved Instances** ile temel kapasiteyi karşılayın ve ek kapasiteyi **Spot Instances** ile sağlayın.
- D. **Reserved Instances** ile temel kapasiteyi karşılayın ve ek kapasiteyi **On-Demand Instances** ile sağlayın.

**Soru Analizi:**

#### Sorunun Gereksinimleri

- Üretim uygulaması:** EC2 örneklerinde çalışıyor.
- Mesaj tabanlı işlem:** Amazon SQS kuyruğundan mesaj okunuyor ve paralel işleniyor.
- Hacim öngörülemez:** Trafik aralıklı ve değişken.
- Kesintisiz işlem:** Uygulama sürekli çalışmalı, mesajlar gecikmemeli veya kaybolmamalı.
- Maliyet etkinlik:** Trafik değişikliklerine rağmen maliyet optimize edilmeli.

Anahtar ifadeler: *değişken trafik, kesintisiz işlem, maliyet etkinlik, EC2 instance türleri*.

**Seçenek Analizi:**

#### C. Reserved Instances + Spot Instances

- Reserved Instances → **temel kapasite** → kesintisiz işleme
- Spot Instances → **artan trafik için ek kapasite** → maliyet düşürür
- Avantaj:** Trafik değişkenliğine uygun, kesintisiz ve maliyet etkin

#### A. Sadece Spot Instances ile maksimum kapasite

- Spot Instances ucuzdur
- Sorun:** Spot Instances herhangi bir anda geri çağrılabılır → kesintisiz işlem garanti edilemez
- Trafik yüksek olsa bile uygulama durabilir → güvenli değil.

#### B. Sadece Reserved Instances ile maksimum kapasite

- Reserved Instances sürekli kullanılabilir → kesintisiz işlem
- Sorun:** Maksimum kapasiteyi her zaman ayırmak maliyetli → değişken trafik için maliyet etkin değil

#### D. Reserved Instances + On-Demand Instances

- Temel kapasiteyi Reserved Instances ile karşılar → kesintisiz işlem ✓
- Artan trafik için On-Demand kullanır → kesintisiz, fakat Spot'a göre daha pahalı !
- **Maliyet etkinlik**, C seçeneğine göre daha düşük.

#### Sonuç

#### Doğru çözüm: C. Reserved Instances ile temel kapasiteyi karşılayıp ek kapasiteyi Spot Instances ile sağlamak ✓

- Kesintisiz işlem garanti edilir
- Trafik değişikliklerine uygun
- Maliyet etkin ve esnek

---

#### QUESTION 38

A security team wants to limit access to specific services or actions in all of the team's AWS accounts. All accounts belong to a large organization in AWS Organizations. The solution must be scalable and there must be a single point where permissions can be maintained.

What should a solutions architect do to accomplish this?

- A. Create an ACL to provide access to the services or actions.
- B. Create a security group to allow accounts and attach it to user groups.
- C. Create cross-account roles in each account to deny access to the services or actions.
- D. Create a service control policy in the root organizational unit to deny access to the services or actions.

#### Soru:

Bir güvenlik ekibi, ekibin sahip olduğu **tüm AWS hesaplarında** belirli **servislere veya işlemlere (actions)** erişimi sınırlamak istiyor. Tüm hesaplar, **AWS Organizations** içinde yer alan büyük bir organizasyona aittir. Çözümün **ölçeklenebilir** olması ve izinlerin **tek bir merkezi noktadan** yönetilebilmesi gerekmektedir.

Bir çözüm mimarı bu hedefe ulaşmak için ne yapmalıdır?

- A. Servislere veya işlemlere erişim sağlamak için bir **ACL** oluşturun.

- B. Hesaplara izin vermek için bir **security group** oluşturun ve bunu kullanıcı gruplarına bağlayın.
- C. Her hesapta, servislere veya işlemlere erişimi engellemek için **cross-account roller** oluşturun.
- D. Servislere veya işlemlere erişimi engellemek için **root organizational unit** üzerinde bir **service control policy (SCP)** oluşturun.

#### Soru Analizi:

##### Sorunun Gereksinimleri

- Birden fazla AWS hesabı:** Tüm hesaplar AWS Organizations altında.
- Belirli servisler veya aksiyonlar kısıtlanacak:** Örneğin belirli AWS servislerinin kullanımı engellenecek.
- Merkezi yönetim:** İzinler tek bir yerden yönetilmeli.
- Ölçeklenebilirlik:** Yeni hesaplar eklendiğinde ek yapılandırma gerekmemeli.

Anahtar ifadeler: *AWS Organizations, merkezi kontrol, ölçeklenebilir izin yönetimi.*

#### Seçenek Analizi:

##### D. Service Control Policy (SCP) oluşturmak

- SCP**, AWS Organizations içinde hesapların **hangi servisleri veya aksiyonları kullanabileceğini** merkezi olarak sınırlar.
- Root Organizational Unit'e uygulanırsa **tüm hesaplara otomatik olarak uygulanır**.
- Tek merkezden yönetim ve yüksek ölçeklenebilirlik sağlar.

##### A. ACL oluşturmak

- ACL'ler genellikle **S3 veya network seviyesinde** kullanılır.
- AWS hesapları genelinde servis veya aksiyon bazlı izinleri yönetmek için uygun değildir.

##### B. Security group oluşturmak

- Security group'lar **network erişimini** kontrol eder (port/IP bazlı).
- AWS servislerine veya aksiyonlara erişimi kısıtlamak için kullanılamaz.

##### C. Cross-account role'ler oluşturmak

- Her hesapta ayrı ayrı rol tanımlamak gereklidir → **ölçeklenebilir değil**.

- Merkezi bir yönetim noktası yoktur. 

### Sonuç

**Doğru çözüm: D. Root organizational unit üzerinde bir Service Control Policy (SCP) oluşturmak **

- Merkezi kontrol
- Tüm hesaplara otomatik etki
- Ölçeklenebilir ve AWS en iyi uygulamalarına uygun

---

### QUESTION 39

A company is concerned about the security of its public web application due to recent web attacks. The application uses an Application Load Balancer (ALB). A solutions architect must reduce the risk of DDoS attacks against the application.

What should the solutions architect do to meet this requirement?

- A. Add an Amazon Inspector agent to the ALB.
- B. Con gure Amazon Macie to prevent attacks.
- C. Enable AWS Shield Advanced to prevent attacks.
- D. Con gure Amazon GuardDuty to monitor the ALB.

#### Soru:

Bir şirket, son zamanlarda gerçekleşen web saldırıları nedeniyle **genel erişime açık web uygulamasının güvenliği** konusunda endişeliidir. Uygulama bir **Application Load Balancer (ALB)** kullanmaktadır. Bir çözüm mimarı, uygulamaya yönelik **DDoS saldırıları riskini azaltmak** zorundadır.

Bu gereksinimi karşılamak için çözüm mimarı ne yapmalıdır?

- A. ALB'ye bir **Amazon Inspector agent** ekleyin.
- B. Saldırıları önlemek için **Amazon Macie** yapılandırın.
- C. Saldırıları önlemek için **AWS Shield Advanced**'i etkinleştirin.
- D. ALB'yi izlemek için **Amazon GuardDuty** yapılandırın.

#### Soru Analizi:

##### Sorunun Gereksinimleri

1. **Genel erişime açık web uygulaması:** İnternete açık ve saldırı yüzeyi yüksek.

2. **Application Load Balancer (ALB)** kullanılıyor.
3. **DDoS saldırılara karşı koruma** isteniyor.
4. **Riskin azaltılması** hedefleniyor (önleme / mitigation).

Anahtar ifade: *DDoS protection, ALB, public web application*.

#### Seçenek Analizi:

##### C. AWS Shield Advanced'i etkinleştirilmek

- **AWS Shield**, DDoS saldırılara karşı koruma sağlar.
- **Shield Advanced**, özellikle **ALB, CloudFront, Route 53** gibi servisler için gelişmiş DDoS koruması sunar.
- Otomatik saldırı tespiti, mitigasyon ve 7/24 DDoS Response Team desteği sağlar.  


##### Amazon Inspector agent eklemek

- Amazon Inspector, **zafiyet taraması** yapar (EC2, container, Lambda).
- **DDoS saldırılarnı önlemez.** 

##### B. Amazon Macie yapılandırmak

- Amazon Macie, **hassas verileri (PII)** tespit etmek için kullanılır.
- DDoS saldırılılarıyla ilgisi yoktur. 

##### D. Amazon GuardDuty yapılandırmak

- GuardDuty, **tehdit tespiti ve izleme** yapar.
- Saldırıları **önlemez**, sadece tespit eder. 

#### Sonuç

Doğru çözüm: C. AWS Shield Advanced'i etkinleştirilmek 

- ALB için yerleşik DDoS koruması
- Gerçek zamanlı saldırı önleme
- Public web uygulamaları için en uygun çözüm

---

#### QUESTION 40

A company's web application is running on Amazon EC2 instances behind an Application Load Balancer. The company recently changed its policy, which now requires the application to be accessed from one specific country only.

Which configuration will meet this requirement?

- A. Configure the security group for the EC2 instances.
- B. Configure the security group on the Application Load Balancer.
- C. Configure AWS WAF on the Application Load Balancer in a VPC.
- D. Configure the network ACL for the subnet that contains the EC2 instances.

#### Soru:

Bir şirketin web uygulaması, **Application Load Balancer (ALB)** arkasında çalışan **Amazon EC2 örnekleri** üzerinde çalışmaktadır. Şirket yakın zamanda politikasını değiştirmiştir ve artık uygulamaya **yalnızca belirli bir ülkeden** erişilmesini zorunlu kılmaktadır.

Bu gereksinimi karşılamak için hangi yapılandırma kullanılmalıdır?

- A. EC2 örnekleri için **security group** yapılandırın.
- B. **Application Load Balancer** üzerindeki **security group**'u yapılandırın.
- C. Bir **VPC** içinde **Application Load Balancer** üzerinde **AWS WAF** yapılandırın.
- D. EC2 örneklerini içeren subnet için **network ACL** yapılandırın.

#### Soru Analizi:

##### Sorunun Gereksinimleri

1. Web uygulaması **ALB arkasında** çalışıyor.
2. Uygulamaya erişim **tek bir ülke ile sınırlanacak**.
3. Çözüm, **coğrafi konuma göre erişim kontrolü** yapabilmeli.
4. Merkezi, yönetilebilir ve uygulama katmanında bir çözüm tercih ediliyor.

Anahtar ifade: *country-based access restriction, ALB, web application*.

#### Seçenek Analizi:

- C. ALB üzerinde AWS WAF yapılandırmak
- **AWS WAF, geo match (ülke bazlı)** kurallar destekler.
  - ALB ile entegre çalışır ve uygulama katmanında trafiği filtreler.

- Politika değişiklikleri merkezi ve kolay yönetilir.

#### A. EC2 security group yapılandırmak

- Security group'lar **IP/CIDR** bazlı çalışır.
- Ülke bazlı (geo-based) kısıtlama yapamaz. 

#### B. ALB security group yapılandırmak

- Yine sadece **IP aralığı** ile kısıtlama yapılabilir.
- Ülke bazlı erişim kontrolü mümkün değildir. 

#### C. Subnet network ACL yapılandırmak

- NACL'ler de IP bazlı çalışır.
- Ülke bazlı kısıtlama desteklemez. 
- Yönetimi karmaşık ve statiktir.

#### Sonuç

#### Doğru çözüm: C. Application Load Balancer üzerinde AWS WAF yapılandırmak

- Ülke bazlı erişim kontrolü sağlar
- ALB ile doğrudan entegre
- Yönetilebilir ve ölçülebilir

---

#### QUESTION 41

A company provides an API to its users that automates inquiries for tax computations based on item prices. The company experiences a larger number of inquiries during the holiday season only that cause slower response times. A solutions architect needs to design a solution that is scalable and elastic.

What should the solutions architect do to accomplish this?

- A. Provide an API hosted on an Amazon EC2 instance. The EC2 instance performs the required computations when the API request is made.
- B. Design a REST API using Amazon API Gateway that accepts the item names. API Gateway passes item names to AWS Lambda for tax computations.
- C. Create an Application Load Balancer that has two Amazon EC2 instances behind it. The EC2 instances will compute the tax on the received item names.

D. Design a REST API using Amazon API Gateway that connects with an API hosted on an Amazon EC2 instance. API Gateway accepts and passes the item names to the EC2 instance for tax computations.

#### Soru:

Bir şirket, ürün fiyatlarına dayalı **vergi hesaplamaları** için sorguları otomatikleştiren bir **API** sunmaktadır. Şirket, yalnızca **tatil sezonunda** çok daha fazla sayıda sorgu almaktadır ve bu durum **yanıt sürelerinin yavaşlamasına** neden olmaktadır. Bir çözüm mimarı, **ölçeklenebilir ve elastik** bir çözüm tasarlamak zorundadır.

Bu gereksinimi karşılamak için çözüm mimarı ne yapmalıdır?

A. Amazon EC2 üzerinde barındırılan bir API sağlayın. API isteği geldiğinde, EC2 örneği gerekli hesaplamaları yapın.

B. Ürün adlarını kabul eden bir **REST API**'yi **Amazon API Gateway** kullanarak tasarlın. API Gateway, ürün adlarını **AWS Lambda**'ya ıletsin ve vergi hesaplamalarını Lambda yapın.

C. Arkasında iki Amazon EC2 örneği bulunan bir **Application Load Balancer** oluşturun. EC2 örnekleri, alınan ürün adları üzerinden vergi hesaplamalarını yapın.

D. Amazon API Gateway kullanarak bir **REST API** tasarlın ve bunu Amazon EC2 üzerinde barındırılan bir API'ye bağlayın. API Gateway, ürün adlarını kabul edip vergi hesaplamaları için EC2 örneğine ıletsin.

#### Soru Analizi:

##### Sorunun Gereksinimleri

- Şirket bir **API** sunuyor (vergi hesaplama).
- Trafik **mevsimsel ve ani artışı** (özellikle tatil sezonunda).
- Mevcut durumda **yanıt süreleri yavaşlıyor** → ölçekleme ihtiyacı var.
- Çözüm **ölçeklenebilir (scalable)** ve **elastik (elastic)** olmalı.
- Operasyonel yük mümkün olduğunca **düşük** olmalı.

Anahtar ifadeler: *API, sezonluk trafik artışı, elastik yapı, otomatik ölçekleme*.

#### Seçenek Analizi:

##### B. API Gateway + AWS Lambda

- API Gateway** otomatik olarak ölçeklenir.
- AWS Lambda**, istek sayısına göre otomatik ölçeklenir → ideal elastik yapı.

- Trafik yalnızca tatil sezonunda artsa bile **boşta maliyet yoktur**.
- Sunucu yönetimi gerekmez (serverless).

### A. API'yi tek bir EC2 instance üzerinde çalıştırma

- Tek EC2 instance → **manuel ölçekleme** gereklidir.
- Trafik arttığında performans düşer.
- Elastik değildir.

### C. ALB + 2 EC2 instance

- Yük dengeleme sağlar ancak:
  - Sadece **iki EC2** → sınırlı kapasite.
  - Auto Scaling belirtilmemiş → elastik değil.
- EC2 yönetimi ve maliyeti devam eder.

### D. API Gateway + EC2

- API Gateway ölçeklenir ama **arka uç EC2 ölçeklenmez**.
- EC2, darboğaz olmaya devam eder.
- Operasyonel yük vardır.

### Sonuç

Amazon API Gateway + AWS Lambda, bu senaryo için en uygun çözümdür çünkü:

- Trafik artışlarına otomatik yanıt verir
- Elastiktir ve serverless mimarı sunar
- Yalnızca kullanım kadar maliyet oluşturur
- Performans sorunlarını ortadan kaldırır

### Sınav ipucu:

Soruda “**API + sezonluk trafik artışı + ölçeklenebilir & elastik**” görürsen → **API Gateway + Lambda** düşün.

### QUESTION 42

A solutions architect is creating a new Amazon CloudFront distribution for an application. Some of the information submitted by users is sensitive. The application

uses HTTPS but needs another layer of security. The sensitive information should be protected throughout the entire application stack, and access to the information should be restricted to certain applications.

Which action should the solutions architect take?

- A. Configure a CloudFront signed URL.
- B. Configure a CloudFront signed cookie.
- C. Configure a CloudFront field-level encryption profile.
- D. Configure CloudFront and set the Origin Protocol Policy setting to HTTPS Only for the Viewer Protocol Policy.

**Soru:**

Bir çözüm mimarı, bir uygulama için yeni bir **Amazon CloudFront dağıtımları (distribution)** oluşturmaktadır. Kullanıcılar tarafından gönderilen bilgilerin bir kısmı **hassastır**. Uygulama **HTTPS** kullanmaktadır ancak **ek bir güvenlik katmanına** ihtiyaç duymaktadır. Hassas bilgilerin **uygulama yığınının (application stack) tamamı boyunca korunması** ve bu bilgilere erişimin **yalnızca belirli uygulamalarla sınırlanması** gerekmektedir.

Bu gereksinimleri karşılamak için çözüm mimarı hangi aksiyonu almalıdır?

- A. **CloudFront signed URL** yapılandırın.
- B. **CloudFront signed cookie** yapılandırın.
- C. **CloudFront field-level encryption profile** yapılandırın.
- D. CloudFront'u yapılandırın ve **Origin Protocol Policy** ayarını **Viewer Protocol Policy için HTTPS Only** olacak şekilde ayarlayın.

**Soru Analizi:**

**Sorunun Gereksinimleri**

1. **Amazon CloudFront** kullanılıyor.
2. Kullanıcıların gönderdiği verilerin bir kısmı **hassas (sensitive)**.
3. Uygulama zaten **HTTPS** kullanıyor → iletişim sırasında şifreleme var.
4. **Ek bir güvenlik katmanı** isteniyor.
5. Hassas veriler, **uygulama yığınının tamamında** (CloudFront → origin → backend) korunmalı.
6. Hassas verilere erişim **yalnızca belirli uygulamalarla** sınırlanılmalı.

Anahtar ifadeler: *sensitive user data, entire application stack, restrict access, extra security layer.*

### Seçenek Analizi:

#### C. CloudFront Field-Level Encryption Profile

- Belirli HTTP alanlarını (ör. form alanları, kredi kartı bilgileri) CloudFront üzerinde şifreler.
- Veri, **origin ve backend dahil tüm stack boyunca şifreli kalır.**
- Şifre çözme yetkisi sadece belirlenen uygulamalara verilir.
- Sorudaki tüm gereksinimleri karşılar.

#### A. CloudFront Signed URL

- Signed URL, içeriğe erişimi zaman/IP bazlı sınırlar.
- **Verinin içeriğini şifrelemez**, sadece kimlerin erişebileceğini kontrol eder.
- Hassas kullanıcı verilerinin korunması için uygun değil. 

#### CloudFront Signed Cookie

- Signed cookie de signed URL gibi **erişim kontrolü** sağlar.
- **Veri alanlarını (fields)** korumaz veya şifrelemez.
- Hassas form verileri için uygun değildir. 

#### D. HTTPS Only (Viewer / Origin Protocol Policy)

- HTTPS zaten kullanılıyor denmiş.
- Bu sadece **taşıma sırasında şifreleme** sağlar.
- Ek güvenlik katmanı veya uygulama içi veri koruması sağlamaz. 

### Sonuç

Doğru cevap: C. CloudFront field-level encryption profile

### Neden?

- Hassas verileri uçtan uca şifreler
- Yalnızca yetkili uygulamalar şifreyi çözebilir
- HTTPS'e ek, uygulama katmanında güvenlik sağlar

### Sınav ipucu

Soruda “**CloudFront + sensitive user data + entire stack boyunca koruma**” görürsen → **Field-Level Encryption** düşün.

---

### QUESTION 43

A gaming company hosts a browser-based application on AWS. The users of the application consume a large number of videos and images that are stored in Amazon S3. This content is the same for all users. The application has increased in popularity, and millions of users worldwide accessing these media files. The company wants to provide the files to the users while reducing the load on the origin.

Which solution meets these requirements MOST cost-effectively?

- A. Deploy an AWS Global Accelerator accelerator in front of the web servers.
- B. Deploy an Amazon CloudFront web distribution in front of the S3 bucket.
- C. Deploy an Amazon ElastiCache for Redis instance in front of the web servers.
- D. Deploy an Amazon ElastiCache for Memcached instance in front of the web servers.

#### Soru:

Bir oyun şirketi, tarayıcı tabanlı bir uygulamayı AWS üzerinde barındırmaktadır. Uygulamanın kullanıcıları, **Amazon S3** üzerinde saklanan çok sayıda **video ve görsel** tüketmektedir. Bu içerikler **tüm kullanıcılar için aynıdır**. Uygulamanın popülerliği artmış ve dünya genelinde **milyonlarca kullanıcı** bu medya dosyalarına erişmektedir. Şirket, dosyaları kullanıcılara sunarken **origin (kaynak) üzerindeki yükü azaltmak** istemektedir.

Bu gereksinimleri **en maliyet etkin** şekilde hangi çözüm karşılar?

- A. Web sunucularının önüne bir **AWS Global Accelerator** yerleştirin.
- B. **Amazon S3 bucket’ın** önüne bir **Amazon CloudFront web distribution** yerleştirin.
- C. Web sunucularının önüne bir **Amazon ElastiCache for Redis** örneği yerleştirin.
- D. Web sunucularının önüne bir **Amazon ElastiCache for Memcached** örneği yerleştirin.

#### Soru Analizi:

##### Sorunun Gereksinimleri

1. **Tarayıcı tabanlı uygulama** → içerik internet üzerinden dağıtılmıyor.
2. **Statik içerik** (video ve görseller) → Amazon S3’te saklanıyor.
3. **İçerik tüm kullanıcılar için aynı**.

4. Dünya genelinde milyonlarca kullanıcı erişiyor.
5. Origin üzerindeki yük azaltılmalı.
6. **En maliyet etkin çözüm** isteniyor.

Anahtar ifadeler: *static content, S3, global users, reduce origin load, cost-effective.*

#### Seçenek Analizi:

##### B. Amazon CloudFront + S3

- CloudFront, **küresel CDN**'dir.
- Video ve görselleri **edge location'larda cache'ler**.
- Kullanıcılar içeriği en yakın edge'ten alır → düşük gecikme.
- **Origin (S3) üzerindeki yükü ciddi şekilde azaltır**.
- Statik, herkes için aynı içerik için **en ideal ve maliyet etkin çözüm**. 

##### A. AWS Global Accelerator

- Global Accelerator, **TCP/UDP tabanlı uygulamalar** için uygundur.
- İçerik **cache'lemez**.
- S3 üzerindeki statik içerik için uygun değildir. 

##### C. ElastiCache for Redis

- Redis, genelde **uygulama verisi / session cache** için kullanılır.
- Medya dosyalarını global ölçekte dağıtmak için uygun değildir. 

##### D. ElastiCache for Memcached

- Memcached de Redis gibi **uygulama içi cache** içindir.
- Büyük medya dosyaları ve global erişim için uygun değildir. 

#### Sonuç

**Doğru cevap: B. Amazon CloudFront web distribution'ı S3 bucket'ın önüne koymak**

#### Neden?

- Statik ve herkes için aynı içerik
- Global kullanıcılar
- Origin yükünü azaltma

- CDN = CloudFront

### Sınav ipucu

Soruda “**S3 + statik içerik + global kullanıcılar + origin load**” görürsen → **CloudFront** düşün.

---

### QUESTION 44

A company has a multi-tier application that runs six front-end web servers in an Amazon EC2 Auto Scaling group in a single Availability Zone behind an Application Load Balancer (ALB). A solutions architect needs to modify the infrastructure to be highly available without modifying the application.

Which architecture should the solutions architect choose that provides high availability?

- A. Create an Auto Scaling group that uses three instances across each of two Regions.
- B. Modify the Auto Scaling group to use three instances across each of two Availability Zones.
- C. Create an Auto Scaling template that can be used to quickly create more instances in another Region.
- D. Change the ALB in front of the Amazon EC2 instances in a round-robin configuration to balance traffic to the web tier.

#### Soru:

Bir şirketin, tek bir **Availability Zone** içinde çalışan ve bir **Application Load Balancer (ALB)** arkasında yer alan, **Amazon EC2 Auto Scaling grubunda** bulunan altı adet ön uç (front-end) web sunucusundan oluşan çok katmanlı (multi-tier) bir uygulaması vardır. Bir **solutions architect**, uygulamada herhangi bir değişiklik yapmadan altyapıyı **yüksek erişilebilir (highly available)** hale getirmek istemektedir.

Bu gereksinimi karşılayan en uygun mimari hangisidir?

- A. İki farklı **Region** genelinde, her birinde üçer instance olacak şekilde bir Auto Scaling grubu oluşturmak.
- B. Auto Scaling grubunu, iki farklı **Availability Zone** genelinde, her birinde üçer instance olacak şekilde yapılandırmak.
- C. Başka bir Region'da hızlıca daha fazla instance oluşturmak için kullanılabilecek bir Auto Scaling şablonu (template) oluşturmak.
- D. Web katmanına gelen trafiği dengelemek için, Amazon EC2 instance'larının önündeki ALB'yi **round-robin** yapılandırmasına geçirmek.

## Soru Analizi:

Soruda verilen **kritik noktalar**:

- Çok katmanlı (multi-tier) bir uygulama
- **6 adet EC2 front-end web server**
- **Auto Scaling Group (ASG)** içinde
- **Tek bir Availability Zone (AZ)**'da çalışıyor
- **Application Load Balancer (ALB)** arkasında
- **Uygulamaya dokunmadan**
- **High Availability (yüksek erişilebilirlik)** isteniyor

### 👉 Ana problem:

Tek AZ kullanıldığı için **AZ seviyesinde bir hata** olursa tüm uygulama erişilemez olur.

### 👉 AWS'te high availability'nin temel kuralı:

**Birden fazla Availability Zone kullanmak**

### 🧠 Doğru Çözüm Mantığı

- Aynı Region içinde
- Birden fazla AZ
- Load Balancer + Auto Scaling ile
- Uygulama değişmeden

**Seçenek Analizi:**

### ✓ B. Auto Scaling grubunu iki Availability Zone'a yaymak

- Mevcut ASG:
  - Tek AZ → riskli
- Çözüm:
  - **2 AZ**
  - Her AZ'de **3 instance**
- ALB:
  - Zaten **multi-AZ çalışabilir**
- Avantajlar:

- AZ down olsa bile uygulama ayakta
- Uygulama değişikliği yok
- AWS best practice

→ **High availability için ideal ve en basit çözüm**

✗ **A. İki Region'da Auto Scaling grubu (Yanlış ✗)**

- Region'lar arası mimari:
  - **Çok pahalı**
  - **Daha karmaşık**
  - Genellikle:
    - DNS (Route 53)
    - Replikasyon
    - Veri senkronizasyonu
    - Uygulama değişiklikleri gerektirir
- Soruda:
  - “without modifying the application” denmiş

→ **Gereksiz ve overkill**

✗ Elenir

✗ **C. Başka Region için Auto Scaling template oluşturmak (Yanlış ✗)**

- Template:
  - Sadece **hazırlık**
  - Gerçek zamanlı HA sağlamaz
- Failover:
  - Manuel
  - Yavaş
- High availability ≠ Disaster recovery hazırlığı

→ **Anında erişilebilirlik sağlama**

✗ Elenir

✗ **D. ALB'yi round-robin yapmak (Yanlış ✗)**

- ALB zaten:
  - Sağlık kontrolü yapar
  - Trafiği otomatik dağıtır
- Round-robin:
  - **AZ hatasını çözmez**
- Problem:
  - Load balancing değil
  - **Tek AZ kullanılması**

→ Yanlış problemi çözüyor

✗ Elenir

🎯 Sonuç

**Auto Scaling grubunu iki Availability Zone'a yayarak, her birinde üç instance çalıştırılmak**

Bu mimari:

- High availability sağlar
- AWS best practice'lere uygundur
- Uygulamaya dokunmaz
- En maliyet-etkin çözümdür

---

#### QUESTION 45

An ecommerce company has an order-processing application that uses Amazon API Gateway and an AWS Lambda function. The application stores data in an Amazon Aurora PostgreSQL database. During a recent sales event, a sudden surge in customer orders occurred. Some customers experienced timeouts, and the application did not process the orders of those customers. A solutions architect determined that the CPU utilization and memory utilization were high on the database because of a large number of open connections. The solutions architect needs to prevent the timeout errors while making the least possible changes to the application.

Which solution will meet these requirements?

- A. Configure provisioned concurrency for the Lambda function. Modify the database to be a global database in multiple AWS Regions.

- B. Use Amazon RDS Proxy to create a proxy for the database. Modify the Lambda function to use the RDS Proxy endpoint instead of the database endpoint.
- C. Create a read replica for the database in a different AWS Region. Use query string parameters in API Gateway to route traffic to the read replica.
- D. Migrate the data from Aurora PostgreSQL to Amazon DynamoDB by using AWS Database Migration Service (AWS DMS). Modify the Lambda function to use the DynamoDB table.

**Soru:**

Bir e-ticaret şirketinin, **Amazon API Gateway** ve **AWS Lambda** kullanan bir **sipariş işleme (order-processing)** uygulaması vardır. Uygulama, verileri **Amazon Aurora PostgreSQL** veritabanında saklamaktadır. Yakın zamanda yapılan bir **satış kampanyası** sırasında, müşteri siparişlerinde ani bir artış yaşanmıştır. Bazı müşteriler **timeout (zaman aşımı)** hatalarıyla karşılaşmış ve bu müşterilerin siparişleri uygulama tarafından işlenmemiştir. Bir **solutions architect**, çok sayıda **açık veritabanı bağlantısı** nedeniyle veritabanında **CPU kullanımının ve bellek kullanımının** çok yüksek olduğunu tespit etmiştir. Solutions architect, **uygulamada mümkün olan en az değişikliği yaparak** bu **timeout hatalarını önlemek** istemektedir.

Bu gereksinimleri karşılayan çözüm hangisidir?

- A. Lambda fonksiyonu için **provisioned concurrency** yapılandırmak. Veritabanını birden fazla AWS Region'da çalışan **global database** haline getirmek.
- B. Veritabanı için bir **Amazon RDS Proxy** oluşturmak. Lambda fonksyonunu, veritabanı endpoint'i yerine **RDS Proxy endpoint'ini** kullanacak şekilde değiştirmek.
- C. Veritabanı için farklı bir AWS Region'da bir **read replica** oluşturmak. API Gateway'de **query string parametreleri** kullanarak trafiği read replica'ya yönlendirmek.
- D. **AWS Database Migration Service (AWS DMS)** kullanarak verileri Aurora PostgreSQL'den **Amazon DynamoDB**'ye taşımak. Lambda fonksyonunu DynamoDB tablosunu kullanacak şekilde değiştirmek.

**Soru Analizi:**

 **Sorunun Özeti**

Bir e-ticaret **sipariş işleme uygulaması** var:

- **API Gateway + AWS Lambda**
- **Veritabanı:** Amazon Aurora PostgreSQL
- **Sorun:**

- Kampanya sırasında **ani trafik artışı**
- Bazı müşteriler **timeout hatası** alıyor
- **Veritabanında çok fazla açık bağlantı (connection)**
- Bu yüzden **CPU ve bellek kullanımı aşırı yükselmiş**

#### Hedef:

- Timeout hatalarını önlemek
- **Uygulamada en az değişiklikle çözüm bulmak**

#### Asıl Problemin Kök Nedeni

Lambda:

- Otomatik ölçeklenir
- Her Lambda invocation → **yeni DB connection açabilir**

Aurora PostgreSQL:

- Aynı anda açılabilecek **connection sayısı sınırlıdır**
- Çok fazla bağlantı →
  - CPU ↑
  - Memory ↑
  - Query'ler yavaşlar → **timeout**

 Yani **asıl sorun connection management**

#### En Doğru Çözüm Yaklaşımı

Lambda + RDS/Aurora kullanıyorsan:

 **Amazon RDS Proxy** tam olarak bu problem için vardır.

Seçenek Analizi:

  **Seçenek B Analizi**

**B. Veritabanı için bir Amazon RDS Proxy oluşturmak.**

Lambda fonksiyonunu, veritabanı endpoint'ı yerine RDS Proxy endpoint'ini kullanacak şekilde değiştirmek.

 **Neden Doğru?**

- **RDS Proxy:**

- DB connection pooling yapar
- Lambda'ların sürekli connection açıp kapatmasını engeller
- Açık connection sayısını ciddi şekilde azaltır
- Veritabanı üzerindeki:
  - CPU kullanımı ↓
  - Memory kullanımı ↓
- **Uygulama değişikliği minimum:**
  - Sadece connection endpoint'i değişir
- AWS'nin **serverless + relational DB** için önerdiği best practice

👉 Bu senaryo birebir **AWS sınavlarında klasik RDS Proxy sorusu**

### ✗ A Seçeneği

Lambda için provisioned concurrency  
Veritabanını global database yapmak

#### Neden yanlış?

- Provisioned concurrency:
  - Cold start problemini çözer
  - **DB connection sayısını azaltmaz**
- Global database:
  - Region-lar arası replikasyon
  - Connection storm problemini çözmez
- Ayrıca:
  - **Çok karmaşık**
  - **Gereksiz**
  - “En az değişiklik” şartına uymaz

### ✗ C Seçeneği

Read replica oluşturmak  
Trafisi query string ile yönlendirmek

#### Neden yanlış?

- Sipariş işleme:
  - Write (**INSERT / UPDATE**) ağırlıklıdır
- Read replica:
  - Write kabul etmez
- API Gateway üzerinden yönlendirme:
  - Karmaşık
  - Application logic değişikliği gerektirir

### D Seçeneği

Aurora → DynamoDB migration

### Neden yanlış?

- Büyük mimari değişiklik
- Tüm data model değişir
- Lambda kodu tamamen değişir
- “**En az değişiklik**” şartına tamamen ters

### Sonuç

### Anahtar Kavram – Sınav İpucu

Soruda şunları görürsen **aklına direkt RDS Proxy gelsin**:

- Lambda + RDS/Aurora
- Timeout
- Too many connections
- High CPU / memory on DB
- Minimal code change

✓ **En az değişiklik**

✓ **Connection management problemi çözülür**

✓ **AWS best practice**

---

### QUESTION 46

An application runs on Amazon EC2 instances in private subnets. The application needs to access an Amazon DynamoDB table.

What is the MOST secure way to access the table while ensuring that the traffic does not leave the AWS network?

- A. Use a VPC endpoint for DynamoDB.
- B. Use a NAT gateway in a public subnet.
- C. Use a NAT instance in a private subnet.
- D. Use the internet gateway attached to the VPC.

#### Soru:

Bir uygulama, **private subnet**'lerde bulunan **Amazon EC2 instance**'ları üzerinde çalışmaktadır. Uygulamanın, bir **Amazon DynamoDB tablosuna erişmesi** gerekmektedir.

Trafiğin **AWS ağı dışına çıkmamasını sağlayarak ve en yüksek güvenliği sunacak** şekilde DynamoDB tablosuna erişmenin yolu hangisidir?

- A. DynamoDB için bir **VPC endpoint** kullanmak.
- B. Public subnet'te bir **NAT gateway** kullanmak.
- C. Private subnet'te bir **NAT instance** kullanmak.
- D. VPC'ye bağlı bir **internet gateway** kullanmak.

#### Soru Analizi:

##### Sorunun Analizi

##### Mevcut Mimari

- Uygulama **Amazon EC2** üzerinde çalışıyor
- EC2'ler **private subnet**'lerde
- Uygulama **Amazon DynamoDB** tablosuna erişmek istiyor

##### Gereksinimler (Soruda özellikle vurgulananlar)

1. **En güvenli yol (MOST secure)**
2. **Trafik AWS ağı dışına çıkmamalı**
3. DynamoDB'ye erişim sağlanmalı

 Bu üç ifade sorunun ana anahtar kelimeleri.

##### Temel AWS Bilgisi

- **DynamoDB**, VPC içinde çalışan bir servis değildir (managed AWS service)
- Normalde internet üzerinden erişilir
- Ancak:
  - **VPC Endpoint (Gateway Endpoint)** kullanılırsa
  - Trafik **AWS backbone network** içinde kalır
  - Internet, NAT veya IGW kullanılmaz

#### Seçenek Analizi:

  A Seçeneği Analizi

#### Use a VPC endpoint for DynamoDB

##### ✓ Neden doğru?

- DynamoDB için kullanılan endpoint türü: **Gateway VPC Endpoint**
- Avantajları:
  - Trafik **AWS ağı dışına çıkmaz**
  - **Internet Gateway** gerekmez
  - **NAT Gateway / NAT Instance** gerekmez
  - En güvenli erişim yöntemidir
- Private subnet'lerdeki EC2'ler:
  - Doğrudan DynamoDB'ye erişebilir

 AWS sınavlarında klasik bilgi:

#### S3 ve DynamoDB → Gateway VPC Endpoint

##### ✗ B. NAT Gateway (Public Subnet)

- NAT Gateway:
  - Trafiği **internet üzerinden** çıkarır
  - AWS backbone dışına çıkarılır
- Güvenli değildir
- Sorudaki:
  - “**traffic does not leave the AWS network**” şartını karşılamaz

##### ✗ C. NAT Instance (Private Subnet)

- NAT instance:
  - Eski (deprecated sayılır)
  - Yine internet üzerinden çıkış sağlar
- Güvenlik ve ölçeklenebilirlik zayıf
- AWS network dışına çıkar

## D. Internet Gateway

- Internet Gateway:
  - Direkt internete çıkış sağlar
- En az güvenli seçenek
- Sorunun ana şartlarına tamamen ters

## Sonuç

### Sınav İpucu (Altın Kural)

Soruda şunları görürsen:

- **Private subnet**
- **S3 veya DynamoDB**
- **Traffic must stay in AWS network**
- **Most secure**

 **VPC Endpoint** → doğru cevap

---

## QUESTION 47

An entertainment company is using Amazon DynamoDB to store media metadata. The application is read intensive and experiencing delays. The company does not have staff to handle additional operational overhead and needs to improve the performance efficiency of DynamoDB without reconfiguring the application.

What should a solutions architect recommend to meet this requirement?

- A. Use Amazon ElastiCache for Redis.
- B. Use Amazon DynamoDB Accelerator (DAX).
- C. Replicate data by using DynamoDB global tables.
- D. Use Amazon ElastiCache for Memcached with Auto Discovery enabled.

### Soru:

Bir eğlence şirketi, medya meta verilerini depolamak için Amazon DynamoDB kullanmaktadır. Uygulama okuma ağırlıklıdır ve gecikmeler yaşamaktadır. Şirketin ek operasyonel yükü yönetecek personeli yoktur ve uygulamayı yeniden yapılandırmadan DynamoDB'nin performans verimliliğini artırması gerekmektedir.

Bu gereksinimi karşılamak için bir solutions architect ne önermelidir?

- A. Amazon ElastiCache for Redis kullanmak.
- B. Amazon DynamoDB Accelerator (DAX) kullanmak.
- C. DynamoDB global tables kullanarak verileri çoğaltmak.
- D. Auto Discovery etkin olacak şekilde Amazon ElastiCache for Memcached kullanmak.

### Soru Analizi:

#### Sorunun Analizi

#### Mevcut Durum

- Veritabanı: **Amazon DynamoDB**
- Veri türü: **Medya metadata**
- Uygulama tipi: **Read-intensive (okuma ağırlıklı)**
- Sorun: **Gecikme (latency)**

#### İş Gereksinimleri (çok kritik)

Soruda özellikle vurgulanan ifadeler:

1. **Read intensive**
2. **Delays (latency problemi)**
3. **No staff for additional operational overhead**
4. **Without reconfiguring the application**

 Bu 4 ifade, doğru çözümü doğrudan işaret ediyor.

#### AWS Bilgisi – Kritik Nokta

- DynamoDB zaten yüksek ölçeklenebilir bir servistir
- Ancak **çok yoğun okuma** durumlarında:
  - Mikro saniye seviyesinde cache gerekebilir
- Ayrıca:
  - **Uygulama kodunda değişiklik istenmiyor**

- Ek operasyonel yük istenmiyor

**Seçenek Analizi:**

  **B Seçeneği Analizi (DOĞRU)**

**Use Amazon DynamoDB Accelerator (DAX)**

**✓ Neden doğru?**

- DAX, DynamoDB için **tamamen yönetilen (fully managed), in-memory cache** servisidir
- Özellikleri:
  - Mikro saniye seviyesinde okuma performansı
  - Read-intensive workload'lar için özel tasarlanmıştır
  - **DynamoDB API ile birebir uyumludur**
- Sonuç:
  - Uygulama **yeniden yapılandırılmaz**
  - Ek operasyonel yük **yok denecek kadar azdır**
  - Gecikmeler ciddi şekilde azalır

 AWS sınav anahtar cümlesi:

*Improve DynamoDB read performance without changing application logic → DAX*

### **✗ A. Amazon ElastiCache for Redis**

- Redis güçlüdür ama:
  - Uygulama kodunda cache logic eklenmesi gereklidir
  - Cache invalidation yönetimi gereklidir
  - Operasyonel yük yüksektir
- Sorudaki:
  - “Without reconfiguring the application”
  - “No additional operational overhead” şartlarını karşılamaz

### **✗ D. ElastiCache for Memcached (Auto Discovery)**

- Redis ile benzer şekilde:

- Uygulama değişikliği gereklidir
- Cache yönetimi gereklidir
- DynamoDB ile **native entegre degildir**
- Yanlış

### C. DynamoDB Global Tables

- Global Tables:
  - **Multi-Region replication** içindir
- Amaç:
  - Düşük latency değil
  - **High availability + disaster recovery**
- Read latency problemini doğrudan çözmez
- Gereksiz ve pahalı

### Sonuç

### Sınav İpuçları (Altın Kurallar)

Soruda şu ifadeleri görürsen:

- DynamoDB
- Read-intensive
- Latency
- No app changes
- Minimal ops

### Refleks cevap: DAX

---

### QUESTION 48

A company's infrastructure consists of Amazon EC2 instances and an Amazon RDS DB instance in a single AWS Region. The company wants to back up its data in a separate Region.

Which solution will meet these requirements with the LEAST operational overhead?

- A. Use AWS Backup to copy EC2 backups and RDS backups to the separate Region.

- B. Use Amazon Data Lifecycle Manager (Amazon DLM) to copy EC2 backups and RDS backups to the separate Region.
- C. Create Amazon Machine Images (AMIs) of the EC2 instances. Copy the AMIs to the separate Region. Create a read replica for the RDS DB instance in the separate Region.
- D. Create Amazon Elastic Block Store (Amazon EBS) snapshots. Copy the EBS snapshots to the separate Region. Create RDS snapshots. Export the RDS snapshots to Amazon S3. Configure S3 Cross-Region Replication (CRR) to the separate Region.

**Soru:**

Bir şirketin altyapısı, tek bir **AWS Region** içinde çalışan **Amazon EC2 instance'ları** ve bir **Amazon RDS veritabanı (DB) instance'ından** oluşmaktadır. Şirket, verilerini **farklı bir Region**'da yedeklemek istemektedir.

Bu gereksinimleri **EN AZ operasyonel yükle (LEAST operational overhead)** karşılayacak çözüm hangisidir?

- A.** AWS Backup kullanarak EC2 yedeklerini ve RDS yedeklerini ayrı bir Region'a kopyalamak.
- B.** Amazon Data Lifecycle Manager (Amazon DLM) kullanarak EC2 yedeklerini ve RDS yedeklerini ayrı bir Region'a kopyalamak.
- C.** EC2 instance'ları için Amazon Machine Image (AMI) oluşturmak ve AMI'leri ayrı bir Region'a kopyalamak. RDS DB instance'ı için ayrı bir Region'da read replica oluşturmak.
- D.** Amazon Elastic Block Store (Amazon EBS) snapshot'ları oluşturmak ve bu snapshot'ları ayrı bir Region'a kopyalamak. RDS snapshot'ları oluşturmak, bu snapshot'ları Amazon S3'e export etmek ve ayrı Region'a Amazon S3 Cross-Region Replication (CRR) yapılandırmak.

**Soru Analizi:**

 **Sorunun Analizi**

 **Mevcut Altyapı**

- **Amazon EC2 instance'ları**
- **Amazon RDS DB instance**
- Hepsi **tek bir AWS Region** içinde

**İş Gereksinimleri**

Soruda özellikle vurgulanan anahtar ifadeler:

1. **Veriler ayrı bir Region'da yedeklenecek**
2. **LEAST operational overhead (en az operasyonel yük)**

👉 Yani çözüm:

- Otomatik olmalı
- Merkezi yönetilebilmeli
- EC2 + RDS için **tek bir çözüm** sunmalı
- Manuel snapshot, kopyalama, karmaşık replikasyon içermemeli

## 🔍 AWS Servisleriyle İlgili Kritik Bilgi

- **AWS Backup:**
  - EC2, EBS, RDS, DynamoDB gibi servisler için
  - **Merkezi, tam yönetilen** backup servisidir
  - **Cross-Region backup copy** desteği vardır
- **Amazon DLM:**
  - Sadece **EBS snapshot** yönetimi içindir
  - RDS'i desteklemez

### Seçenek Analizi:

✓ A Seçeneği Analizi

**Use AWS Backup to copy EC2 backups and RDS backups to the separate Region.**

### ✓ Neden doğru?

- AWS Backup:
  - EC2 ve RDS yedeklerini **tek yerden yönetir**
  - Otomatik backup planları oluşturur
  - **Cross-Region copy** özelliği vardır
- Avantajlar:
  - En az operasyonel yük
  - Manuel işlem yok
  - AWS tarafından yönetilir
- Sorudaki tüm gereksinimleri **doğrudan karşılar**

📌 Sınav anahtar cümlesi:

*Centralized backup + multiple AWS services + least operational overhead → AWS Backup*

## B. Amazon Data Lifecycle Manager (DLM)

- DLM:
  - **Sadece EBS snapshot'ları** yönetir
  - RDS backup'larını desteklemez
- EC2 + RDS birlikte yönetilemez
- Gereksinimi **tam karşılamaz**

## C. AMI + RDS Read Replica

- AMI:
  - EC2 için kullanılabilir
  - Manuel işlem ve yönetim gerektirir
- RDS Read Replica:
  - Backup değil
  - Sürekli çalışan ek DB instance → maliyet
- **Operational overhead yüksektir**
- Yanlış amaçlı servis kullanımı

## D. EBS Snapshot + S3 Export + CRR

- Çok fazla adım içerir:
  - EBS snapshot
  - Snapshot copy
  - RDS snapshot export
  - S3 CRR
- Yönetimi karmaşık
- Operasyonel yük **en yüksek** seçenek
- Sorudaki “least operational overhead” şartına ters

## Sonuç

## Sınav İpuçları (Altın Kurallar)

Soruda şu ifadeleri görürsen:

- EC2 + RDS

- Backup
- Cross-Region
- Least operational overhead

👉 AWS Backup refleks cevap olmalı

---

## QUESTION 49

A solutions architect needs to securely store a database user name and password that an application uses to access an Amazon RDS DB instance. The application that accesses the database runs on an Amazon EC2 instance. The solutions architect wants to create a secure parameter in AWS Systems Manager Parameter Store.

What should the solutions architect do to meet this requirement?

- A. Create an IAM role that has read access to the Parameter Store parameter. Allow Decrypt access to an AWS Key Management Service (AWS KMS) key that is used to encrypt the parameter. Assign this IAM role to the EC2 instance.
- B. Create an IAM policy that allows read access to the Parameter Store parameter. Allow Decrypt access to an AWS Key Management Service (AWS KMS) key that is used to encrypt the parameter. Assign this IAM policy to the EC2 instance.
- C. Create an IAM trust relationship between the Parameter Store parameter and the EC2 instance. Specify Amazon RDS as a principal in the trust policy.
- D. Create an IAM trust relationship between the DB instance and the EC2 instance. Specify Systems Manager as a principal in the trust policy.

**Soru:**

Bir **solutions architect**, bir uygulamanın **Amazon RDS DB instance**'ına erişmek için kullandığı **veritabanı kullanıcı adı ve parolasını** güvenli bir şekilde saklamak istemektedir. Uygulama, **Amazon EC2 instance** üzerinde çalışmaktadır. Solutions architect, **AWS Systems Manager Parameter Store** içinde **güvenli (secure)** bir **parametre** oluşturmak istemektedir.

Bu gereksinimi karşılamak için solutions architect ne yapmalıdır?

- A. Parameter Store parametresine okuma erişimi olan bir **IAM rolü** oluşturmak. Parametreyi şifrelemek için kullanılan **AWS Key Management Service (AWS KMS)** anahtarına **Decrypt** erişimi vermek. Bu IAM rolünü **EC2 instance'a atamak**.
- B. Parameter Store parametresine okuma erişimi veren bir **IAM policy** oluşturmak. Parametreyi şifrelemek için kullanılan **AWS Key Management Service (AWS KMS)** anahtarına **Decrypt** erişimi vermek. Bu IAM policy'yi **EC2 instance'a atamak**.

- C.** Parameter Store parametresi ile EC2 instance arasında bir **IAM trust relationship** oluşturmak. Trust policy'de **Amazon RDS**'yi principal olarak belirtmek.
- D.** DB instance ile EC2 instance arasında bir **IAM trust relationship** oluşturmak. Trust policy'de **Systems Manager**'ı principal olarak belirtmek.

#### Soru Analizi:

##### Sorunun Analizi

##### Senaryo

- Bir uygulama:
  - **Amazon EC2** üzerinde çalışıyor
  - **Amazon RDS** veritabanına bağlıyor
- Saklanacak bilgi:
  - **DB kullanıcı adı ve parola (secret)**
- Kullanılacak servis:
  - **AWS Systems Manager Parameter Store**
  - **SecureString** (KMS ile şifrelenmiş)
- Gereksinim:
  - **Güvenli erişim**
  - **Doğru IAM yetkilendirme modeli**

#### Kritik Gereksinimler (Anahtar Kelimeler)

Sorudaki önemli ifadeler:

1. **Securely store**
2. **Secure parameter (Parameter Store – SecureString)**
3. **Application runs on EC2**
4. **Least complexity / correct access method**

 AWS'de EC2 → AWS servis erişimi **IAM Role** ile yapılır

 IAM policy **doğrudan EC2'ye atanamaz**

##### AWS Güvenlik Modeli – Bilinmesi Gereken

- **Parameter Store SecureString:**
  - KMS ile şifrelenir

- Okumak için:
  - ssm:GetParameter
  - kms:Decrypt izni gereklidir
- **EC2 instance:**
  - AWS hizmetlerine erişim için **IAM Role** kullanılır
  - Role → Instance Profile → EC2

#### **Seçenek Analizi:**

##### A Seçeneği Analizi (DOĞRU)

Create an IAM role that has read access to the Parameter Store parameter. Allow Decrypt access to an AWS KMS key that is used to encrypt the parameter. Assign this IAM role to the EC2 instance.

#### ✓ Neden doğru?

- IAM Role:
  - EC2 için **en güvenli ve önerilen yöntem**
  - Credential'lar EC2 içinde saklanmaz
- Gerekli izinler:
  - ssm:GetParameter
  - kms:Decrypt
- AWS best practice:
  - **EC2 → IAM Role → AWS hizmetleri**

##### Sınav ipucu:

*EC2 accessing AWS service securely → IAM Role*

#### B. IAM Policy'yi EC2'ye atamak

- IAM policy:
  - **Tek başına EC2'ye atanamaz**
  - Policy → Role veya User'a eklenir
- Mimari olarak **geçersiz**
- Sınavda klasik tuzak

## C. Parameter Store ile EC2 arasında trust relationship

- Parameter Store:
  - IAM principal değildir
- Trust relationship:
  - **Role ↔ AWS service** arasında olur
- Amazon RDS'nin burada rolü yok
- Kavramsal olarak yanlış

## D. DB instance ile EC2 arasında trust relationship

- RDS:
  - IAM principal değildir (IAM auth hariç)
- Systems Manager:
  - Principal olarak bu senaryoda kullanılmaz
- Yanlış servis ilişkisi

### Sonuç

### Sınav İpuçları (Altın Kurallar)

Soruda şunları görürsen:

- EC2
- Secrets / credentials
- Parameter Store
- Secure access

### IAM Role + KMS Decrypt refleks cevap olmalı

---

## QUESTION 50

A company is designing a cloud communications platform that is driven by APIs. The application is hosted on Amazon EC2 instances behind a Network Load Balancer (NLB). The company uses Amazon API Gateway to provide external users with access to the application through APIs. The company wants to protect the platform against web exploits like SQL injection and also wants to detect and mitigate large, sophisticated DDoS attacks.

Which combination of solutions provides the MOST protection? (Choose two.)

- A. Use AWS WAF to protect the NLB.
- B. Use AWS Shield Advanced with the NLB.
- C. Use AWS WAF to protect Amazon API Gateway.
- D. Use Amazon GuardDuty with AWS Shield Standard
- E. Use AWS Shield Standard with Amazon API Gateway.

**Soru:**

Bir şirket, API'ler tarafından yönlendirilen bir bulut iletişim platformu tasarlamaktadır. Uygulama, bir Network Load Balancer (NLB) arkasında çalışan Amazon EC2 instance'ları üzerinde barındırılmaktadır. Şirket, dış kullanıcıların API'ler aracılığıyla uygulamaya erişmesini sağlamak için Amazon API Gateway kullanmaktadır. Şirket, platformu SQL injection gibi web tabanlı saldırırlara karşı korumak ve büyük, karmaşık DDoS saldırısını tespit edip azaltmak istemektedir.

EN FAZLA korumayı sağlayan çözüm kombinasyonu hangisidir? (İki seçenek seçiniz.)

- A. Network Load Balancer'ı korumak için AWS WAF kullanmak.
- B. Network Load Balancer ile birlikte AWS Shield Advanced kullanmak.
- C. Amazon API Gateway'ı korumak için AWS WAF kullanmak.
- D. AWS Shield Standard ile birlikte Amazon GuardDuty kullanmak.
- E. Amazon API Gateway ile birlikte AWS Shield Standard kullanmak.

**Soru Analizi:**

 **Sorunun Analizi**

 **Mimari**

- **Amazon EC2** → arka planda uygulama
- **Network Load Balancer (NLB)** → EC2'lerin önünde
- **Amazon API Gateway** → dış kullanıcıların API erişimi
- Platform: **API-driven**

**Güvenlik Gereksinimleri (çok kritik)**

Soruda iki ayrı tehdit açıkça belirtiliyor:

1. **Web exploit'ler (SQL injection gibi)**
2. **Large, sophisticated DDoS attacks**
3. **MOST protection** (en yüksek seviye koruma)
4. **İki çözüm seçilecek**

👉 Bu şu anlama gelir:

- **Uygulama katmanı (Layer 7)** koruması gereklidir
- **Ağ katmanı (Layer 3/4)** için güçlü DDoS koruması gereklidir

## 🔍 AWS Servisleri – Kritik Bilgiler

### AWS WAF

- **Layer 7 (HTTP/HTTPS)** koruma
- SQL injection, XSS gibi saldırıları engeller
- Şu servislerle entegre çalışır:
  - **Amazon API Gateway**
  - Application Load Balancer
  - CloudFront
- ❌ **NLB ile çalışmaz**

### AWS Shield

- **DDoS koruması**
- İki tür:
  - **Shield Standard** → otomatik, temel
  - **Shield Advanced** → büyük ve karmaşık saldırılar için

📌 “Large, sophisticated DDoS attacks” ifadesi → **Shield Advanced**

### Seçenek Analizi:

✓ **B Seçeneği Analizi**

### Use AWS Shield Advanced with the NLB

#### ✓ Neden doğru?

- NLB:
  - Layer 4 (TCP/UDP)
  - Büyük hacimli DDoS saldırılarına maruz kalabilir
- Shield Advanced:
  - Gelişmiş DDoS algılama
  - Otomatik mitigation

- AWS DDoS Response Team (DRT) desteği
- “Large, sophisticated DDoS” gereksinimini **en iyi karşılayan servis**

## C Seçeneği Analizi

### Use AWS WAF to protect Amazon API Gateway

#### ✓ Neden doğru?

- API Gateway:
  - HTTP tabanlıdır
- AWS WAF:
  - SQL injection
  - XSS
  - Web exploit'ler
- Sorudaki:
  - “protect against web exploits like SQL injection” gereksinimini **doğrudan karşılar**

#### A. AWS WAF ile NLB'yi korumak

- AWS WAF:
  - NLB'yi **desteklemez**
- Teknik olarak mümkün değil

#### D. GuardDuty + Shield Standard

- GuardDuty:
  - **Threat detection**
  - Mitigation yapmaz
- Shield Standard:
  - Temel DDoS koruması
- “MOST protection” şartını karşılamaz

#### E. Shield Standard + API Gateway

- Shield Standard:
  - Zaten API Gateway için **default olarak aktiftir**

- Ek koruma sağlamsız
- Büyük, sofistike saldırılardan yetersiz

### Sonuç

#### Sınav İpuçları (Altın Kurallar)

Soruda şu ifadeleri görürsen:

- **SQL injection / web exploits** → AWS WAF
  - **Large, sophisticated DDoS** → AWS Shield Advanced
  - **API Gateway** → WAF uygulanabilir
  - **NLB** → Shield Advanced
- 

### QUESTION 51

A company has a legacy data processing application that runs on Amazon EC2 instances. Data is processed sequentially, but the order of results does not matter. The application uses a monolithic architecture. The only way that the company can scale the application to meet increased demand is to increase the size of the instances. The company's developers have decided to rewrite the application to use a microservices architecture on Amazon Elastic Container Service (Amazon ECS).

What should a solutions architect recommend for communication between the microservices?

- Create an Amazon Simple Queue Service (Amazon SQS) queue. Add code to the data producers, and send data to the queue. Add code to the data consumers to process data from the queue.
- Create an Amazon Simple Notification Service (Amazon SNS) topic. Add code to the data producers, and publish notifications to the topic. Add code to the data consumers to subscribe to the topic.
- Create an AWS Lambda function to pass messages. Add code to the data producers to call the Lambda function with a data object. Add code to the data consumers to receive a data object that is passed from the Lambda function.
- Create an Amazon DynamoDB table. Enable DynamoDB Streams. Add code to the data producers to insert data into the table. Add code to the data consumers to use the DynamoDB Streams API to detect new table entries and retrieve the data.

**Soru:**

Bir şirketin, **Amazon EC2 instance'ları** üzerinde çalışan **eski (legacy)** bir veri işleme uygulaması vardır. Veriler **sıralı (sequential)** olarak işlenmektedir, ancak **sonuçların sırası önemli değildir**. Uygulama **monolitik bir mimarı** kullanmaktadır. Şirketin, artan talebi karşılamak için uygulamayı ölçekleyebilmesinin **tek yolu instance boyutunu artırmaktır (scale up)**. Şirketin geliştiricileri, uygulamayı **Amazon Elastic Container Service (Amazon ECS)** üzerinde **mikroservis mimarisi** kullanacak şekilde yeniden yazmaya karar vermiştir.

Bir solutions architect, **mikroservisler arasındaki iletişim** için ne önermelidir?

- A. Bir **Amazon Simple Queue Service (Amazon SQS)** kuyruğu oluşturmak. Veri üreticilere (data producers) kuyruğa veri gönderecek şekilde kod eklemek ve veri tüketicilere (data consumers) kuyrukta veri işleyebilecek şekilde kod eklemek.
- B. Bir **Amazon Simple Notification Service (Amazon SNS)** konusu (topic) oluşturmak. Veri üreticilere bu konuya bildirim yayınılayacak şekilde kod eklemek ve veri tüketicilere bu konuya abone olacak şekilde kod eklemek.
- C. Mesajları iletmek için bir **AWS Lambda fonksiyonu** oluşturmak. Veri üreticilere veri nesnesiyle birlikte Lambda fonksyonunu çağıracak şekilde kod eklemek ve veri tüketicilere Lambda fonksyonundan iletilen veri nesnesini alacak şekilde kod eklemek.
- D. Bir **Amazon DynamoDB** tablosu oluşturmak ve **DynamoDB Streams**'i etkinleştirmek. Veri üreticilere tabloya veri ekleyecek şekilde kod eklemek ve veri tüketicilere DynamoDB Streams API'sini kullanarak yeni tablo kayıtlarını algılayıp verileri alacak şekilde kod eklemek.

#### Soru Analizi:

##### Sorunun Analizi

##### Mevcut Durum

- Uygulama:
  - **Legacy**
  - **Monolithic**
  - **EC2 üzerinde**
- Veri işleme:
  - **Sequential**
  - **Sonuç sırası önemli değil**
- Ölçekleme problemi:
  - Sadece **scale up** yapılabiliyor (instance büyütme)
  - Yatay ölçekleme yok

## Hedef Mimari

- **Microservices**
- **Amazon ECS**
- **Yatay ölçeklenebilir**
- Servisler arasında:
  - **Loose coupling (gevşek bağılılık)**
  - **Asenkron iletişim**

👉 “Order does not matter” ifadesi **çok kritik**

👉 Bu, **queue-based** bir tasarımlı doğrudan işaret eder

## 🔍 AWS Servisleri – Kritik Eşleşmeler

### Amazon SQS

- Mesaj tabanlı
- Asenkron
- Order önemli değilse **Standard Queue**
- Producer-consumer modeline çok uygundur
- Microservices için **best practice**

### Amazon SNS

- **Pub/Sub**
- Fan-out senaryoları için
- Aynı mesajın **birden fazla tüketiciye** gitmesi gereklidir
- Bu senaryoda gerek yok

### AWS Lambda (aracı olarak)

- Gereksiz ara katman
- Ek gecikme
- Ek maliyet
- Microservice iletişimini için önerilmez

### DynamoDB + Streams

- Event-driven tasarım için

- Mesajlaşma sistemi değildir
- Karmaşıklık ve ek maliyet getirir

### Seçenek Analizi:

#### A Seçeneği Analizi

### Amazon SQS queue kullanmak

#### ✓ Neden doğru?

- SQS:
  - Producer ve consumer'ları **decouple eder**
  - Asenkron çalışır
  - ECS servisleriyle doğal uyumludur
- Order önemli değil:
  - **Standard Queue** tam uyumlu
- Sonuç:
  - Kolay yatay ölçekleme
  - Microservices mimarisine uygun
  - Düşük operasyonel yük

#### Sınav anahtar cümlesi:

*Microservices + async + order not important → SQS*

#### B. Amazon SNS

- SNS:
  - Push tabanlı
  - Birden fazla subscriber içindir
- Bu senaryoda:
  - Tek consumer grubu yeterli
- Yanlış servis tipi

#### C. AWS Lambda aracı

- Gereksiz mimari karmaşıklık
- Lambda microservice communication için **anti-pattern**

- Ölçeklenebilir ama anlamsız

## D. DynamoDB + Streams

- Streams:
  - Değişiklik yakalama (CDC)
- Messaging için kullanmak:
  - Over-engineering
- Yanlış kullanım

## Sonuç

### Sınav İpuçları (Altın Kurallar)

Soruda şu ifadeleri görürsen:

- Legacy → Microservices
- ECS
- Sequential ama order önemli değil
- Scale out ihtiyacı

## Refleks cevap: SQS

---

## QUESTION 52

A company wants to migrate its MySQL database from on premises to AWS. The company recently experienced a database outage that significantly impacted the business. To ensure this does not happen again, the company wants a reliable database solution on AWS that minimizes data loss and stores every transaction on at least two nodes.

Which solution meets these requirements?

- A. Create an Amazon RDS DB instance with synchronous replication to three nodes in three Availability Zones.
- B. Create an Amazon RDS MySQL DB instance with Multi-AZ functionality enabled to synchronously replicate the data.
- C. Create an Amazon RDS MySQL DB instance and then create a read replica in a separate AWS Region that synchronously replicates the data.

D. Create an Amazon EC2 instance with a MySQL engine installed that triggers an AWS Lambda function to synchronously replicate the data to an Amazon RDS MySQL DB instance.

#### Soru:

Bir şirket, şirket içi (on-premises) ortamdan MySQL veritabanını AWS'ye taşımak istemektedir. Şirket kısa süre önce, iş faaliyetlerini önemli ölçüde etkileyen bir veritabanı kesintisi yaşamıştır. Bunun tekrar yaşanmamasını sağlamak için şirket, AWS üzerinde güvenilir bir veritabanı çözümü istemektedir. Bu çözüm, veri kaybını en aza indirmeli ve her işlemi en az iki düğümde saklamalıdır.

Bu gereksinimleri karşılayan çözüm hangisidir?

- A. Üç farklı Availability Zone'da bulunan üç düğüme senkron replikasyon yapan bir Amazon RDS DB instance oluşturmak.
- B. Multi-AZ özelliği etkinleştirilmiş bir Amazon RDS MySQL DB instance oluşturarak verileri senkron olarak çoğaltmak.
- C. Bir Amazon RDS MySQL DB instance oluşturmak ve ardından verileri senkron olarak çoğaltan, farklı bir AWS Region'da bir read replica oluşturmak.
- D. Üzerinde MySQL çalıştırılan bir Amazon EC2 instance oluşturmak ve verileri senkron olarak çoğaltmak için bir AWS Lambda fonksiyonu tetiklemek; verileri bir Amazon RDS MySQL DB instance'ına kopyalamak.

#### Soru Analizi:

##### Senaryo

- Kaynak sistem: **On-premises MySQL**
- Hedef: **AWS**
- Geçmiş problem:
  - **Ciddi veritabanı kesintisi (outage)**
- Yeni bekleni:
  - **Yüksek güvenilirlik**
  - **Minimum veri kaybı**
  - **Her transaction en az iki node'da saklanmalı**

 Bu ifadeler AWS tarafından **synchronous replication + high availability** ihtiyacını açıkça gösterir.

#### Kritik Gereksinimler (Anahtar Kelimeler)

1. **Minimize data loss**

2. Every transaction stored on at least two nodes

3. Reliable managed solution

👉 AWS terminolojisinde bu = Multi-AZ, synchronous replication

### 🔍 AWS Bilgisi – Önemli Noktalar

#### Amazon RDS Multi-AZ

- Primary + Standby instance
- Farklı Availability Zone'larda
- Senkron replikasyon
- Failover otomatik
- Transaction commit:
  - Primary ve Standby yazıldıktan sonra tamamlanır

📌 Bu tam olarak sorunun istediği davranıştır.

#### Seçenek Analizi:

✓ B Seçeneği Analizi

Multi-AZ özelliği etkinleştirilmiş bir Amazon RDS MySQL DB instance

#### ✓ Neden doğru?

- Senkron replikasyon:
  - Her işlem **en az iki node'da** tutulur
- Veri kaybı riski:
  - **En aza indirilir**
- Yüksek erişilebilirlik:
  - Otomatik failover
- Operasyonel yük:
  - AWS tarafından yönetilir
- MySQL uyumluluğu:
  - On-prem MySQL → RDS MySQL geçişi kolay

#### ✗ A. 3 node, 3 AZ, senkron replikasyon

- RDS:

- Native olarak **3 node synchronous replication** sunmaz (Aurora hariç)
- Teknik olarak geçersiz / hatalı tanım
- Yanlıtıcı seçenek

### C. Cross-Region read replica (synchronous)

- Read replica:
  - **Asenkron** çalışır
- Cross-Region:
  - Latency yüksek
  - Veri kaybı mümkündür
- “Minimize data loss” şartını karşılamaz

### D. EC2 + MySQL + Lambda replikasyonu

- Custom replikasyon:
  - Karmaşık
  - Güvenilir değil
  - Operasyonel yük çok yüksek
- AWS managed çözüm değil
- Sınavda “anti-pattern”

### Sonuç

### Sınav İpuçları (Altın Kurallar)

Soruda şu ifadeleri görürsen:

- MySQL
- High availability
- Minimize data loss
- Synchronous replication

### Refleks cevap: RDS Multi-AZ

---

## QUESTION 53

A company is building a new dynamic ordering website. The company wants to minimize server maintenance and patching. The website must be highly available and must scale read and write capacity as quickly as possible to meet changes in user demand.

Which solution will meet these requirements?

- A. Host static content in Amazon S3. Host dynamic content by using Amazon API Gateway and AWS Lambda. Use Amazon DynamoDB with on-demand capacity for the database. Configure Amazon CloudFront to deliver the website content.
- B. Host static content in Amazon S3. Host dynamic content by using Amazon API Gateway and AWS Lambda. Use Amazon Aurora with Aurora Auto Scaling for the database. Configure Amazon CloudFront to deliver the website content.
- C. Host all the website content on Amazon EC2 instances. Create an Auto Scaling group to scale the EC2 instances. Use an Application Load Balancer to distribute traffic. Use Amazon DynamoDB with provisioned write capacity for the database.
- D. Host all the website content on Amazon EC2 instances. Create an Auto Scaling group to scale the EC2 instances. Use an Application Load Balancer to distribute traffic. Use Amazon Aurora with Aurora Auto Scaling for the database.

**Soru:**

Bir şirket, yeni ve dinamik bir **sipariş verme (ordering)** web sitesi geliştirmektedir. Şirket, **sunucu bakımı ve patch (güncelleme)** işlemlerini en aza indirmek istemektedir. Web sitesi **yüksek erişilebilir** olmalı ve **kullanıcı talebindeki değişimlere hızlıca yanıt verebilmek için okuma ve yazma kapasitesini mümkün olan en hızlı şekilde ölçektebilmelidir.**

Bu gereksinimleri karşılayan çözüm hangisidir?

- A. Statik içeriği **Amazon S3** üzerinde barındırmak. Dinamik içeriği **Amazon API Gateway** ve **AWS Lambda** kullanarak sunmak. Veritabanı olarak **on-demand capacity** kullanan **Amazon DynamoDB**'yi kullanmak. Web sitesi içeriğini sunmak için **Amazon CloudFront** yapılandırmak.
- B. Statik içeriği **Amazon S3** üzerinde barındırmak. Dinamik içeriği **Amazon API Gateway** ve **AWS Lambda** kullanarak sunmak. Veritabanı olarak **Aurora Auto Scaling** kullanan **Amazon Aurora**'yı kullanmak. Web sitesi içeriğini sunmak için **Amazon CloudFront** yapılandırmak.
- C. Web sitesinin tüm içeriğini **Amazon EC2 instance'ları** üzerinde barındırmak. EC2 instance'larını ölçektelemek için bir **Auto Scaling group** oluşturmak. Trafiği dağıtmak için bir **Application Load Balancer** kullanmak. Veritabanı olarak **provisioned write capacity** kullanan **Amazon DynamoDB**'yi kullanmak.

D. Web sitesinin tüm içeriğini **Amazon EC2 instance'ları** üzerinde barındırmak. EC2 instance'larını ölçeklemek için bir **Auto Scaling group** oluşturmak. Trafiği dağıtmak için bir **Application Load Balancer** kullanmak. Veritabanı olarak **Aurora Auto Scaling** kullanan **Amazon Aurora**'yı kullanmak.

#### Soru Analizi:

##### İş Gereksinimleri (çok kritik)

Soruda özellikle vurgulanan ifadeler:

1. **Minimize server maintenance and patching**
2. **Highly available**
3. **Scale read and write capacity as quickly as possible**
4. **Dynamic ordering website** (okuma + yazma yoğun)

 Bu ifadeler bize şunu söyler:

- **Serverless / managed servisler** tercih edilmeli
- EC2 tabanlı çözümler **istenmiyor**
- Anı trafik artışlarına **otomatik ve hızlı ölçeklenme** gerekli
- Veritabanı da **anında ölçeklenebilir** olmalı

##### AWS Servisleri – Kritik Eşleşmeler

#### Server Maintenance & Patching

-  EC2 → manuel bakım, patch
-  Lambda, API Gateway, DynamoDB → **AWS tarafından yönetilir**

#### Hızlı Ölçeklenme (Read + Write)

- **DynamoDB on-demand:**
  - Anında ölçeklenir
  - Provisioning gerekmez
- Aurora:
  - Ölçeklenebilir ama:
    - Instance tabanlı
    - Ölçeklenme süresi saniyeler–dakikalar

#### Seçenek Analizi:

  **A Seçeneği Analizi**

**S3 + API Gateway + Lambda + DynamoDB on-demand + CloudFront**

**✓ Neden doğru?**

- **S3:**
  - Statik içerik
  - Bakım yok
- **API Gateway + Lambda:**
  - Serverless
  - Otomatik ölçeklenir
  - Patch/bakım yok
- **DynamoDB on-demand:**
  - Read & write kapasitesi **anında ölçeklenir**
  - Trafik tahmini gerekmeyez
- **CloudFront:**
  - Global cache
  - Yüksek erişilebilirlik

 Bu mimari:

- Tamamen **serverless**
- **En az operasyonel yük**
- **En hızlı ölçeklenme**

 **B. Aurora Auto Scaling kullanmak**

- Aurora:
  - Yönetilen DB ama:
    - Instance tabanlı
    - Ölçeklenme süresi var
- “Scale as quickly as possible” gereksinimine:
  - DynamoDB kadar hızlı yanıt veremez
- Yanlış değil ama **en iyi çözüm değil**

## C. EC2 + ALB + DynamoDB provisioned

- EC2:
  - Sunucu bakımı ve patch gereklidir
- DynamoDB provisioned:
  - Kapasite önceden ayarlanmalıdır
  - Anı trafik artışlarında throttle riski
- Gereksinimlere ters

## D. EC2 + ALB + Aurora

- En yüksek operasyonel yük
- En yavaş ölçeklenen mimari
- Sorunun tam tersi

### Sonuç

### Sınav İpuçları (Altın Kurallar)

Soruda şu ifadeleri görürsen:

- **Minimize maintenance**
- **Scale as quickly as possible**
- **Highly available**
- **Dynamic web app**

 **Serverless + DynamoDB on-demand** refleks cevap olmalı

---

## QUESTION 54

A company has an AWS account used for software engineering. The AWS account has access to the company's on-premises data center through a pair of AWS Direct Connect connections. All non-VPC traffic routes to the virtual private gateway.

A development team recently created an AWS Lambda function through the console. The development team needs to allow the function to access a database that runs in a private subnet in the company's data center.

Which solution will meet these requirements?

A. Configure the Lambda function to run in the VPC with the appropriate security group.

- B. Set up a VPN connection from AWS to the data center. Route the traffic from the Lambda function through the VPN.
- C. Update the route tables in the VPC to allow the Lambda function to access the on-premises data center through Direct Connect.
- D. Create an Elastic IP address. Configure the Lambda function to send traffic through the Elastic IP address without an elastic network interface.

**Soru:**

Bir şirketin, yazılım mühendisliği için kullanılan bir **AWS hesabı** vardır. Bu AWS hesabı, şirketin **on-premises (şirket içi) veri merkezine** bir çift **AWS Direct Connect** bağlantısı aracılığıyla erişime sahiptir. **VPC dışına giden tüm trafik, virtual private gateway'ye** yönlendirilmiştir. Bir geliştirme ekibi yakın zamanda **AWS Management Console** üzerinden bir **AWS Lambda fonksiyonu** oluşturmuştur. Geliştirme ekibinin, bu fonksiyonun şirketin veri merkezindeki **private subnet**'te çalışan bir **veritabanına erişmesini** sağlaması gerekmektedir.

Bu gereksinimleri karşılayan çözüm hangisidir?

- A. Lambda fonksiyonunu, uygun bir **security group** ile birlikte **VPC içinde çalışacak** şekilde yapılandırmak.
- B. AWS'den veri merkezine bir **VPN bağlantısı** kurmak ve Lambda fonksiyonundan gelen trafiği VPN üzerinden yönlendirmek.
- C. Lambda fonksiyonunun **Direct Connect** üzerinden şirket içi veri merkezine erişmesine izin vermek için **VPC route table**'larını güncellemek.
- D. Bir **Elastic IP address** oluşturmak ve Lambda fonksiyonunu, bir **elastic network interface** kullanmadan trafiği bu Elastic IP üzerinden gönderecek şekilde yapılandırmak.

**Soru Analizi:**

 **Sorunun Analizi**

 **Mevcut Mimari**

- AWS hesabı:
  - Şirketin **on-premises veri merkezine**
  - **AWS Direct Connect** ile bağlı
- Ağ yapılandırması:
  - **VPC dışına giden tüm trafik → Virtual Private Gateway**
- On-premises taraf:
  - **Private subnet**'te çalışan bir **veritabanı**

- Yeni durum:
  - **AWS Lambda fonksiyonu** oluşturulmuş
- Gereksinim:
  - Lambda'nın **on-premises private subnet'teki DB'ye erişmesi**

### **Kritik AWS Bilgisi (Lambda + Network)**

- Varsayılan olarak:
  - **Lambda fonksiyonları VPC dışında çalışır**
- Lambda'nın:
  - VPC içindeki kaynaklara
  - On-premises (Direct Connect / VPN üzerinden) erişebilmesi için:
    - 👉 **VPC içine yerleştirilmesi gereklidir**
- Bu işlem:
  - Lambda'ya **Elastic Network Interface (ENI)** ekler
  - **Security Group** ile trafik kontrol edilir

 Direct Connect zaten mevcut → ekstra bağlantı gerekmez

### **Anahtar Gereksinimler**

Sorudaki belirleyici ifadeler:

1. **On-premises private subnet**
2. **Direct Connect zaten var**
3. **Lambda'nın erişmesi gerekiyor**
4. **En doğru ve basit çözüm**

### **Seçenek Analizi:**

  **A Seçeneği Analizi**

**Lambda fonksiyonunu, uygun bir security group ile birlikte VPC içinde çalışacak şekilde yapılandırmak**

### **✓ Neden doğru?**

- Lambda VPC içine alındığında:
  - VPC route table'larını kullanır

- Direct Connect → Virtual Private Gateway → On-prem DB yolunu izler
- Security Group:
  - DB portu (ör. 3306, 5432) açılır
- Ek altyapı gerekmez
- AWS best practice

📌 Sınav refleksi:

*Lambda → On-prem / VPC resource → “Run Lambda in a VPC”*

### ✖ B. VPN bağlantısı kurmak

- Direct Connect zaten mevcut
- VPN:
  - Gereksiz
  - Ek karmaşıklık
- Soruda **yeni bağlantı istenmiyor**

### ✖ C. Route table güncellemek

- Lambda **VPC dışında çalışıyorsa**:
  - Route table'lar **etkili değildir**
- Önce Lambda'nın VPC içine alınması gereklidir
- Eksik ve yanlış çözüm

### ✖ D. Elastic IP kullanmak

- Lambda:
  - Varsayılan olarak Elastic IP kullanamaz
  - ENI olmadan outbound IP atanamaz
- Teknik olarak mümkün değil
- Sınavda “bariz yanlış” seçenek

🎯 Sonuç

### 🧠 Sınav İpuçları (Altın Kurallar)

Soruda şu kombinasyonu görürsen:

- Lambda

- VPC / On-prem / Direct Connect
- Private subnet erişimi

👉 Refleks cevap: Lambda'yı VPC içine al

---

## QUESTION 55

A company runs an application using Amazon ECS. The application creates resized versions of an original image and then makes Amazon S3 API calls to store the resized images in Amazon S3.

How can a solutions architect ensure that the application has permission to access Amazon S3?

- Update the S3 role in AWS IAM to allow read/write access from Amazon ECS, and then relaunch the container.
- Create an IAM role with S3 permissions, and then specify that role as the taskRoleArn in the task definition.
- Create a security group that allows access from Amazon ECS to Amazon S3, and update the launch configuration used by the ECS cluster.
- Create an IAM user with S3 permissions, and then relaunch the Amazon EC2 instances for the ECS cluster while logged in as this account.

### Soru:

Bir şirket, **Amazon ECS** kullanarak bir uygulama çalıştırmaktadır. Uygulama, bir orijinal görüntünün yeniden boyutlandırılmış (resized) sürümlerini oluşturur ve ardından bu yeniden boyutlandırılmış görüntüleri **Amazon S3**'te depolamak için **Amazon S3 API çağrıları** yapar.

Bir solutions architect, uygulamanın **Amazon S3'e erişim iznine sahip olmasını** nasıl sağlayabilir?

- AWS IAM'deki **S3 rolünü**, Amazon ECS'ten okuma/yazma erişimine izin verecek şekilde güncellemek ve ardından container'ı yeniden başlatmak.
- S3 izinlerine sahip bir IAM rolü** oluşturmak ve bu rolü **task definition** içinde **taskRoleArn** olarak belirtmek.
- Amazon ECS'ten Amazon S3'e erişime izin veren bir **security group** oluşturmak ve bu security group'u ECS cluster'ı tarafından kullanılan **launch configuration**'da güncellemek.
- S3 izinlerine sahip bir **IAM user** oluşturmak ve ardından bu hesapla oturum açarak ECS cluster'ı için kullanılan **Amazon EC2 instance**'larını yeniden başlatmak.

## Soru Analizi:

### Sorunun Analizi

### Senaryo

- Uygulama:
  - **Amazon ECS** üzerinde çalışıyor
  - Container'lar:
    - GörSELLERI yeniden boyutlandırıyor
    - **Amazon S3 API çağrıları** yapıyor
- Gereksinim:
  - Uygulamanın **S3'e erişim izni** olması

 Bu, klasik bir **ECS task → AWS service erişimi** senaryosudur.

### AWS ECS + IAM Temel Bilgisi

- ECS'de iki farklı IAM rolü vardır:
  1. **Task execution role**
    - Image pull (ECR), CloudWatch Logs gibi işlemler
  2. **Task role (taskRoleArn)**
    - **Uygulamanın AWS servislerine erişimi** için kullanılır

 S3 API çağrısı yapan **uygulama kodu** → **task role** kullanır

## Anahtar Gereksinimler

- **En güvenli ve doğru yöntem**
- Hard-coded credential yok
- Container-level yetkilendirme

## Seçenek Analizi:

### B Seçeneği Analizi

**S3 izinlerine sahip bir IAM rolü oluşturmak ve bu rolü task definition'da taskRoleArn olarak belirtmek**

### Neden doğru?

- IAM Role:

- Short-term credentials sağlar
- Güvenlidir
- taskRoleArn:
  - Container içindeki uygulama bu rolü kullanır
- AWS best practice:
  - **ECS → IAM Role → AWS servisleri**

📌 Sınav refleksi:

*ECS application accessing AWS service → taskRoleArn*

#### ✗ A. S3 rolünü ECS için güncellemek

- S3 tarafında “ECS’e izin vermek” diye bir kavram yok
- IAM yetkilendirme yanlış anlaşılmış
- Yanıltıcı seçenek

#### ✗ C. Security group ile S3 erişimi

- Security group:
  - **Network-level**
- S3:
  - IAM ile korunur, SG ile değil
- Yanlış güvenlik katmanı

#### ✗ D. IAM user ile EC2’leri yeniden başlatmak

- IAM user:
  - Long-term credentials
  - **Best practice değil**
- Container bazlı yetkilendirme sağlamaz
- Güvenlik riski

🎯 Sonuç

#### 🧠 Sınav İpuçları (Altın Kurallar)

Soruda şunları görürsen:

- ECS

- Application → S3 API call
- Permission / access

👉 Refleks cevap: IAM Role + taskRoleArn

---

## QUESTION 56

A company has a Windows-based application that must be migrated to AWS. The application requires the use of a shared Windows file system attached to multiple Amazon EC2 Windows instances that are deployed across multiple Availability Zone:

What should a solutions architect do to meet this requirement?

- A. Configure AWS Storage Gateway in volume gateway mode. Mount the volume to each Windows instance.
- B. Configure Amazon FSx for Windows File Server. Mount the Amazon FSx file system to each Windows instance.
- C. Configure a file system by using Amazon Elastic File System (Amazon EFS). Mount the EFS file system to each Windows instance.
- D. Configure an Amazon Elastic Block Store (Amazon EBS) volume with the required size. Attach each EC2 instance to the volume. Mount the file system within the volume to each Windows instance.

### Soru:

Bir şirketin, **AWS'ye taşınması gereken Windows tabanlı bir uygulaması** vardır. Uygulama, **birden fazla Availability Zone'a dağıtılmış birden fazla Amazon EC2 Windows instance'ına bağlı, paylaşımı bir Windows dosya sistemi** kullanılmasını gerektirmektedir.

Bir solutions architect bu gereksinimi karşılamak için ne yapmalıdır?

- A. **AWS Storage Gateway'ı volume gateway** modunda yapılandırmak. Bu volume'ü her bir Windows instance'a mount etmek.
- B. **Amazon FSx for Windows File Server** yapılandırmak. Amazon FSx dosya sistemini her bir Windows instance'a mount etmek.
- C. **Amazon Elastic File System (Amazon EFS)** kullanarak bir dosya sistemi yapılandırmak. EFS dosya sistemini her bir Windows instance'a mount etmek.
- D. Gerekli boyutta bir **Amazon Elastic Block Store (Amazon EBS)** volume'ü yapılandırmak. Her bir EC2 instance'ını bu volume'ye bağlamak. Volume içindeki dosya sistemini her bir Windows instance'a mount etmek.

### Soru Analizi:

## Sorunun Analizi

### Temel Gereksinimler

Soruda özellikle vurgulanan kritik noktalar:

1. Windows tabanlı uygulama
2. Paylaşımı (shared) dosya sistemi
3. Birden fazla EC2 Windows instance
4. Birden fazla Availability Zone
5. Aynı dosya sistemine eş zamanlı erişim

 Bu, klasik bir **Windows shared file system (SMB)** ihtiyacıdır.

### AWS Servis Bilgisi (Anahtar)

Servis	Windows	Multi-AZ Paylaşımı Protokol		
FSx for Windows	✓	✓	✓	SMB
EFS	✗ (Windows native değil)	✓	✓	NFS
EBS	✗	✗	✗	Block
Storage Gateway	✗ (Bu kullanım için)	✗	✗	Hibrit

 Windows uygulamalarında **SMB (Server Message Block)** gereklidir

 AWS'de bunun managed karşılığı: **Amazon FSx for Windows File Server**

**Seçenek Analizi:**

  **B Seçeneği Analizi (DOĞRU)**

**Amazon FSx for Windows File Server yapılandırmak ve dosya sistemini her Windows instance'a mount etmek**

 **Neden doğru?**

- Native **Windows file system**
- **SMB protokolü**
- **Multi-AZ high availability**
- Birden fazla EC2 instance aynı anda erişebilir
- Windows uygulamalarıyla tam uyumlu

 Sınav refleksi:

*Windows + shared file system + multiple AZ → FSx for Windows*

### A. AWS Storage Gateway (Volume Gateway)

- Daha çok:
  - On-premises ↔ AWS entegrasyonu
- EC2'ler arası shared Windows file system için uygun değil
- Multi-AZ ve SMB shared kullanım sağlamaz

### C. Amazon EFS

- Linux için tasarlanmıştır
- **NFS tabanlıdır**
- Windows uygulamaları için **native destek yok**
- Soruda açıkça **Windows file system** isteniyor

### D. Amazon EBS

- **Block storage**
- Aynı anda birden fazla EC2 instance'a bağlanamaz
- AZ bağımlıdır (multi-AZ değil)
- Shared file system değildir

 Sonuç

 **Sınav İpuçları (Altın Kurallar)**

Soruda şu kelimeleri görürsen:

- Windows
- Shared file system
- Multiple EC2
- Multiple AZ

 **FSx for Windows File Server**

---

**QUESTION 57**

A company is developing an ecommerce application that will consist of a load-balanced front end, a container-based application, and a relational database. A solutions architect needs to create a highly available solution that operates with as little manual intervention as possible.

Which solutions meet these requirements? (Choose two.)

- A. Create an Amazon RDS DB instance in Multi-AZ mode.
- B. Create an Amazon RDS DB instance and one or more replicas in another Availability Zone.
- C. Create an Amazon EC2 instance-based Docker cluster to handle the dynamic application load.
- D. Create an Amazon Elastic Container Service (Amazon ECS) cluster with a Fargate launch type to handle the dynamic application load.
- E. Create an Amazon Elastic Container Service (Amazon ECS) cluster with an Amazon EC2 launch type to handle the dynamic application load.

**Soru:**

Bir şirket, **yük dengelemeli (load-balanced)** bir ön yüz, **container tabanlı bir uygulama** ve **ilişkisel bir veritabanından** oluşanak bir **e-ticaret uygulaması** geliştirmektedir. Bir solutions architect'in, **yüksek erişilebilirliğe (high availability) sahip ve mümkün olan en az manuel müdahale ile çalışan** bir çözüm oluşturması gerekmektedir.

Bu gereksinimleri karşılayan çözümler hangileridir? (**İki tane seçin.**)

- A. Multi-AZ modunda bir **Amazon RDS DB instance** oluşturmak.
- B. Bir **Amazon RDS DB instance** ve başka bir **Availability Zone**'da bir veya daha fazla **replica** oluşturmak.
- C. Dinamik uygulama yükünü yönetmek için **Amazon EC2 instance tabanlı bir Docker cluster** oluşturmak.
- D. Dinamik uygulama yükünü yönetmek için **Fargate launch type** kullanan bir **Amazon Elastic Container Service (Amazon ECS) cluster** oluşturmak.
- E. Dinamik uygulama yükünü yönetmek için **Amazon EC2 launch type** kullanan bir **Amazon Elastic Container Service (Amazon ECS) cluster** oluşturmak.

**Soru Analizi:**



- Load-balanced front end

- Container tabanlı uygulama
- Relational database

### Temel Gereksinimler

1. **High availability (yüksek erişilebilirlik)**
2. **As little manual intervention as possible**  
→ Sunucu yönetimi, patching, kapasite yönetimi minimum olmalı

 Bu iki ifade sorunun en kritik noktalarıdır.

### AWS Tasarım Mantığı

#### Katman En Uygun Yaklaşım

Database Managed + HA

Container Serverless / managed

Compute Otomatik ölçekleme

 **Managed servisler**, manuel müdahaleyi azaltır.

#### Seçenek Analizi:

  A Seçeneği Analizi (DOĞRU)

#### Multi-AZ modunda Amazon RDS DB instance

##### ✓ Neden doğru?

- Primary + standby farklı AZ'lerde
- **Synchronous replication**
- Otomatik failover
- Yüksek erişilebilirlik
- DBA müdahalesi minimum

 Sınav refleksi:

*Relational DB + high availability → RDS Multi-AZ*

  D Seçeneği Analizi (DOĞRU)

#### ECS cluster with Fargate launch type

##### ✓ Neden doğru?

- Sunucu yönetimi yok (no EC2 management)
- Otomatik ölçekleme
- Patch, OS, capacity AWS tarafından yönetilir
- En az manuel müdahale

📌 Sınav refleksi:

*Containers + minimal ops → ECS Fargate*

✗ **B. RDS + replica başka AZ'de**

- Read replica:
  - Read scalability için
  - **HA çözümü değildir**
- Failover otomatik değil
- Multi-AZ yerine geçmez

✗ **C. EC2 tabanlı Docker cluster**

- EC2:
  - OS patching
  - Capacity management
- Manuel operasyon yükü yüksek

✗ **E. ECS EC2 launch type**

- ECS var ama:
  - EC2 instance yönetimi hâlâ sizin
- Fargate'e göre daha fazla operasyonel yük

⌚ Sonuç

🧠 **Sınav İpuçları**

Soruda şu ifadeleri görürsen:

- **High availability**
- **Least manual intervention**
- **Container-based**

## 👉 RDS Multi-AZ + ECS Fargate

---

### QUESTION 58

A company uses Amazon S3 as its data lake. The company has a new partner that must use SFTP to upload data files. A solutions architect needs to implement a highly available SFTP solution that minimizes operational overhead.

Which solution will meet these requirements?

- A. Use AWS Transfer Family to configure an SFTP-enabled server with a publicly accessible endpoint. Choose the S3 data lake as the destination.
- B. Use Amazon S3 File Gateway as an SFTP server. Expose the S3 File Gateway endpoint URL to the new partner. Share the S3 File Gateway endpoint with the new partner.
- C. Launch an Amazon EC2 instance in a private subnet in a VPC. Instruct the new partner to upload files to the EC2 instance by using a VPN. Run a cron job script, on the EC2 instance to upload files to the S3 data lake.
- D. Launch Amazon EC2 instances in a private subnet in a VPC. Place a Network Load Balancer (NLB) in front of the EC2 instances. Create an SFTP listener port for the NLB. Share the NLB hostname with the new partner. Run a cron job script on the EC2 instances to upload files to the S3 data lake.

**Soru:**

Bir şirket, **Amazon S3'ü veri gölü (data lake)** olarak kullanmaktadır. Şirketin, **SFTP kullanarak veri dosyaları yüklemesi gereken yeni bir iş ortağı** vardır. Bir solutions architect'in, **yüksek erişilebilirliğe sahip ve operasyonel yükü en aza indiren bir SFTP çözümü** uygulaması gerekmektedir.

Bu gereksinimleri karşılayan çözüm hangisidir?

- A. AWS Transfer Family** kullanarak, **SFTP etkinleştirilmiş ve herkese açık (publicly accessible) bir endpoint**'e sahip bir sunucu yapılandırmak. Hedef olarak **S3 data lake'i** seçmek.
- B. Amazon S3 File Gateway'i** bir **SFTP sunucusu** olarak kullanmak. S3 File Gateway endpoint URL'sini yeni iş ortağına açmak ve paylaşmak.
- C. Bir VPC içindeki private subnet'te** bir **Amazon EC2 instance** başlatmak. Yeni iş ortağına, dosyaları **VPN kullanarak** bu EC2 instance'a yüklemesini söylemek. EC2 instance üzerinde çalışan bir **cron job script** ile dosyaları S3 data lake'e yüklemek.
- D. Bir VPC içindeki private subnet'te** birden fazla **Amazon EC2 instance** başlatmak. Bu instance'ların önüne bir **Network Load Balancer (NLB)** koymak. NLB üzerinde bir **SFTP**

**listener portu** oluşturmak. NLB hostname'ini yeni iş ortağıyla paylaşmak. EC2 instance'lar üzerinde çalışan **cron job script**'leri ile dosyaları S3 data lake'e yüklemek.

### Soru Analizi:



#### Sorunun Analizi



#### Temel Gereksinimler

Soruda özellikle vurgulanan kritik noktalar:

1. **Amazon S3 data lake**
2. **Yeni iş ortağı SFTP kullanmak zorunda**
3. **High availability (yüksek erişilebilirlik)**
4. **Minimum operational overhead (en az operasyonel yük)**

👉 Bu, klasik bir **managed SFTP → S3** entegrasyonu senaryosudur.



#### AWS'de Doğru Servis

İhtiyaç	En Uygun AWS Servisi
---------	----------------------

SFTP	<b>AWS Transfer Family</b>
------	----------------------------

S3 entegrasyonu **Native**

High availability **Managed service**

Minimum ops **No EC2 / no cron**

👉 AWS Transfer Family:

- **SFTP, FTPS, FTP** destekler
- Backend olarak **Amazon S3** veya **Amazon EFS**
- AWS tarafından yönetilir (patch, scale, HA)

### Seçenek Analizi:



#### A Seçeneği Analizi (DOĞRU)

**AWS Transfer Family ile SFTP endpoint oluşturmak ve hedef olarak S3 seçmek**

#### ✓ Neden doğru?

- SFTP doğrudan desteklenir
- S3 data lake ile native entegrasyon

- Public endpoint → partner erişimi kolay
- EC2 yok → operasyonel yük minimum
- AWS tarafından otomatik ölçeklenir ve HA'dır

📌 Sınav refleksi:

*SFTP + S3 + low ops → AWS Transfer Family*

## ✖ B. S3 File Gateway

- File Gateway:
  - NFS / SMB için
  - SFTP desteklemez
- Yanlış protokol

## ✖ C. EC2 + VPN + cron

- EC2 yönetimi:
  - Patch
  - Scaling
- VPN:
  - Ek operasyon
- Cron job:
  - Manual ve kırılgan
- **High availability yok**

## ✖ D. EC2 + NLB + cron

- Daha karmaşık
- EC2 yönetimi hâlâ var
- HA için ASG gereklidir (yok)
- Cron job → operasyonel risk

🎯 Sonuç

## 🧠 Sınav İpuçları (Altın Kurallar)

Soruda şu ifadeleri görürsen:

- Partner

- SFTP
- S3
- Least operational overhead

## 👉 AWS Transfer Family

### Sınav Kuralı

SFTP + S3 + low ops → AWS Transfer Family

---

### QUESTION 59

A company needs to store contract documents. A contract lasts for 5 years. During the 5-year period, the company must ensure that the documents cannot be overwritten or deleted. The company needs to encrypt the documents at rest and rotate the encryption keys automatically every year.

Which combination of steps should a solutions architect take to meet these requirements with the LEAST operational overhead? (Choose two.)

- A. Store the documents in Amazon S3. Use S3 Object Lock in governance mode.
- B. Store the documents in Amazon S3. Use S3 Object Lock in compliance mode.
- C. Use server-side encryption with Amazon S3 managed encryption keys (SSE-S3). Configure key rotation.
- D. Use server-side encryption with AWS Key Management Service (AWS KMS) customer managed keys. Configure key rotation.
- E. Use server-side encryption with AWS Key Management Service (AWS KMS) customer provided (imported) keys. Configure key rotation.

### Soru:

Bir şirketin **sözleşme belgelerini** saklaması gerekmektedir. Bir sözleşme **5 yıl** sürdürmektedir. Bu **5 yıllık süre boyunca**, şirket belgelerin **üzerine yazılamasını (overwrite)** veya **silinememesini** garanti etmelidir. Şirket ayrıca belgelerin **dinlenme halinde (at rest) şifrelenmesini** ve **şifreleme anahtarlarının her yıl otomatik olarak döndürülmesini (rotate)** istemektedir.

Bir solutions architect, **en az operasyonel yük** ile bu gereksinimleri karşılamak için aşağıdaki adımlardan hangilerini seçmelidir? (**İki tane seçin.**)

- A. Belgeleri **Amazon S3**'te saklamak. **S3 Object Lock'u governance mode**'da kullanmak.
- B. Belgeleri **Amazon S3**'te saklamak. **S3 Object Lock'u compliance mode**'da

kullanmak.

- C. Amazon S3 tarafından yönetilen şifreleme anahtarları (SSE-S3) ile server-side encryption kullanmak. Key rotation yapılandırmak.
- D. AWS Key Management Service (AWS KMS) tarafından yönetilen customer managed key'ler ile server-side encryption kullanmak. Key rotation yapılandırmak.
- E. AWS KMS customer provided (imported) key'leri ile server-side encryption kullanmak. Key rotation yapılandırmak.

#### Soru Analizi:



Soruda özellikle vurgulanan kritik noktalar:

1. **Belgeler 5 yıl boyunca silinemez ve üzerine yazılamaz**
2. **Yasal / sözleşmesel koruma (immutability)**
3. **Encryption at rest**
4. **Encryption key'leri her yıl otomatik rotate edilmeli**
5. **En az operasyonel yük**

👉 Bu, klasik bir **immutable + encrypted object storage** senaryosudur.



#### Gereksinim

#### AWS Servisi

Silinemez / değiştirilemez veri **S3 Object Lock (Compliance Mode)**

Encryption at rest

**SSE-KMS**

Otomatik key rotation

**KMS Customer Managed Key**

Düşük operasyonel yük

**Managed services**

#### Seçenek Analizi:



#### S3 Object Lock – Compliance Mode

##### ✓ Neden doğru?

- **Compliance mode:**
  - **Hiç kimse**, root dahil, nesneleri silemez veya değiştiremez

- Retention süresi dolmadan değişiklik mümkün değil
- Governance mode:
  - Özel izinlerle bypass edilebilir → **yeterli değil**

📌 Sınav refleksi:

*Legal / regulatory immutability → Compliance mode*

#### ✓ D Seçeneği Analizi

### SSE-KMS (Customer Managed Key) + automatic rotation

#### ✓ Neden doğru?

- Yalnızca **customer managed CMK'ler**:
  - Otomatik yıllık rotation destekler
- SSE-S3:
  - Key rotation kontrolü yok
- Imported key:
  - Rotation **desteklemez**

📌 Sınav refleksi:

*Auto key rotation → KMS customer managed key*

#### ✗ A. Object Lock – Governance Mode

- Privileged kullanıcılar override edebilir
- **5 yıl garanti silinemezlik sağlanamaz**

#### ✗ C. SSE-S3

- AWS managed key
- **Rotation yapılandırılamaz**
- Key lifecycle kontrolü yok

#### ✗ E. KMS imported keys

- Otomatik rotation **desteklenmez**
- Manuel yönetim gereklidir → yüksek ops

⌚ Sonuç

## Sınav İpuçları (Altın Kurallar)

Soruda şu ifadeleri görürsen:

- **Cannot be deleted or overwritten**
- **Legal / contract documents**
- **Key rotation automatically**

### S3 Object Lock (Compliance) + SSE-KMS (CMK)

#### S3 Object Lock Nedir?

S3 Object Lock, nesnelerin:

- **Silinmesini**
- **Üzerine yazılmasını**  
belirli bir süre boyunca **tamamen engeller**.

#### Özellikle:

- Sözleşmeler
- Finansal kayıtlar
- Yasal belgeler  
için kullanılır.

### Compliance Mode Neden Gerekli?

Object Lock'un iki modu vardır:

#### Mod              Özellik

Governance Yetkili kullanıcılar bypass edebilir

**Compliance Hiç kimse (root dahil) silemez**

#### Soruda:

“5 yıl boyunca silinemez veya overwrite edilemez”

Bu ifade **kesintikle Compliance Mode** demektir.

### Nasıl Çalışır?

1. **Bucket oluşturulurken** Object Lock aktif edilir  
(sonradan açılamaz ! )
2. Nesneler yükleniğinde:

- Retention period (örneğin **5 yıl**) atanır

3. Retention süresi dolmadan:

- Delete ✗
- Overwrite ✗
- Retention değiştirme ✗

### Avantajı

- Yasal uyumluluk (SEC, FINRA, ISO vb.)
- İnsan hatasına karşı koruma
- Ek yazılım gerekmez

### Encryption at Rest – SSE-KMS (Customer Managed Key)

#### 🔒 Neden SSE-KMS?

Soruda:

“Encryption keys automatically rotate every year”

Bu gereksinim **yalnızca KMS customer managed key** ile sağlanır.

#### 🔑 KMS Key Türleri Karşılaştırma

Tür	Otomatik Rotation
SSE-S3	✗
KMS AWS-managed	✗
<b>KMS Customer-managed</b>	✓
KMS Imported key	✗

#### ⌚ Otomatik Key Rotation Nasıl Çalışır?

- AWS KMS:
  - Her yıl **yeni bir key version** üretir
  - Eski veriler:
    - Eski key version ile decrypt edilir
    - Uygulama değişikliği gerekmez
- **Tamamen AWS tarafından yönetilir**

📌 Operasyonel yük **yok** denecek kadar az

### Bu İki Özelliğin Birlikte Çalışması

#### Gereksinim      Çözüm

5 yıl silinemez      S3 Object Lock (Compliance)

Overwrite engeli      S3 Object Lock

Encryption at rest SSE-KMS

Yıllık key rotation      KMS CMK

Low ops      Fully managed

---

### QUESTION 60

A company has a web application that is based on Java and PHP. The company plans to move the application from on premises to AWS. The company needs the ability to test new site features frequently. The company also needs a highly available and managed solution that requires minimum operational overhead.

Which solution will meet these requirements?

- A. Create an Amazon S3 bucket. Enable static web hosting on the S3 bucket. Upload the static content to the S3 bucket. Use AWS Lambda to process all dynamic content.
- B. Deploy the web application to an AWS Elastic Beanstalk environment. Use URL swapping to switch between multiple Elastic Beanstalk environments for feature testing.
- C. Deploy the web application to Amazon EC2 instances that are configured with Java and PHP. Use Auto Scaling groups and an Application Load Balancer to manage the website's availability.
- D. Containerize the web application. Deploy the web application to Amazon EC2 instances. Use the AWS Load Balancer Controller to dynamically route traffic between containers that contain the new site features for testing.

#### Soru:

Bir şirketin **Java ve PHP tabanlı** bir **web uygulaması** vardır. Şirket, uygulamayı **on-premises ortamdan AWS'ye taşımayı** planlamaktadır. Şirketin, **yeni site özelliklerini sık sık test edebilme** ihtiyacı vardır. Ayrıca şirket, **yüksek erişilebilirliğe sahip, yönetilen (managed)** ve **minimum operasyonel yük** gerektiren bir çözüm istemektedir.

Bu gereksinimleri karşılayan çözüm hangisidir?

- A.** Bir **Amazon S3 bucket** oluşturmak. S3 bucket üzerinde **static web hosting**'i etkinleştirmek. Statik içeriği S3 bucket'a yüklemek. Tüm dinamik içeriği işlemek için **AWS Lambda** kullanmak.
- B.** Web uygulamasını bir **AWS Elastic Beanstalk environment**'ına dağıtmak. Özellik (feature) testleri için birden fazla Elastic Beanstalk environment arasında geçiş yapmak amacıyla **URL swapping** kullanmak.
- C.** Web uygulamasını **Java ve PHP** ile yapılandırılmış **Amazon EC2 instance**'larına dağıtmak. Web sitesinin erişilebilirliğini yönetmek için **Auto Scaling group**'lar ve bir **Application Load Balancer** kullanmak.
- D.** Web uygulamasını **container** haline getirmek. Uygulamayı **Amazon EC2 instance**'larına dağıtmak. Test amacıyla yeni site özelliklerini içeren container'lar arasında trafiği dinamik olarak yönlendirmek için **AWS Load Balancer Controller** kullanmak.

**Soru Analizi:**

 **Sorunun Analizi**

 **Uygulamanın Özellikleri**

- **Java ve PHP tabanlı** web uygulaması
- Şu an **on-premises**
- AWS'ye **lift & modernize** edilecek

 **Temel Gereksinimler**

1. **Yeni özellikleri sık sık test edebilme**
2. **High availability**
3. **Managed solution**
4. **Minimum operational overhead**

 **Sorunun kilit kelimeleri:**

**frequently test, managed, minimum ops**

 **AWS Servis Mantığı**

<b>İhtiyaç</b>	<b>En Uygun Çözüm</b>
----------------	-----------------------

Java + PHP runtime	Elastic Beanstalk
--------------------	-------------------

Blue/green deployment	URL swapping
-----------------------	--------------

HA	Beanstalk (ALB + ASG)
----	-----------------------

İhtiyaç	En Uygun Çözüm
---------	----------------

Low ops	Managed PaaS
---------	--------------

### Seçenek Analizi:

#### B Seçeneği Analizi

#### Elastic Beanstalk + URL swapping

##### ✓ Neden doğru?

- Java ve PHP **native destek**
- AWS:
  - EC2
  - ALB
  - Auto Scaling
  - Monitoring  
→ hepsini **otomatik yönetir**
- **URL swapping:**
  - Blue/Green deployment
  - Sık feature test için ideal
- En az operasyonel yük

##### Sınav refleksi:

*Java/PHP + frequent testing + low ops → Elastic Beanstalk*

#### A. S3 + Lambda

- Java/PHP **tam web uygulaması** için uygun değil
- Dinamik içerik tamamı Lambda'ya taşınamaz
- Refactor gereklidir → yüksek efor

#### C. EC2 + ASG + ALB

- HA sağlar ama:
  - OS patching
  - Runtime yönetimi
  - Deployment süreçleri manuel

- **Managed değil**

## D. Containers + EC2

- Container orchestration:
  - ECS/EKS gerektirir
- EC2 yönetimi devam eder
- Ops yükü yüksek

### Sonuç

### Sınav İpuçları (Ezber)

Soruda şunları görürsen:

- **Java / PHP**
- **Frequently test**
- **Minimum ops**
- **Managed**

### Elastic Beanstalk + URL swapping

#### Sınav Ezberi

**Java/PHP + frequent testing + low ops = Elastic Beanstalk**

---

#### QUESTION 61

A company has an ordering application that stores customer information in Amazon RDS for MySQL. During regular business hours, employees run one-time queries for reporting purposes. Timeouts are occurring during order processing because the reporting queries are taking a long time to run. The company needs to eliminate the timeouts without preventing employees from performing queries.

What should a solutions architect do to meet these requirements?

- A. Create a read replica. Move reporting queries to the read replica.
- B. Create a read replica. Distribute the ordering application to the primary DB instance and the read replica.
- C. Migrate the ordering application to Amazon DynamoDB with on-demand capacity.
- D. Schedule the reporting queries for non-peak hours.

**Soru:**

Bir şirketin, müşteri bilgilerini **Amazon RDS for MySQL** üzerinde saklayan bir **sipariş (ordering) uygulaması** vardır. Normal çalışma saatleri sırasında, çalışanlar **raporlama amacıyla tek seferlik (one-time) sorgular** çalıştırmaktadır. Bu raporlama sorgularının uzun süre çalışması nedeniyle, **sipariş işleme sırasında timeout (zaman aşımı) hataları** oluşmaktadır. Şirket, **çalışanların sorgu çalıştırmasını engellemeden**, bu timeout sorunlarını ortadan kaldırmak istemektedir.

Bu gereksinimleri karşılamak için bir solutions architect ne yapmalıdır?

- A. Bir **read replica** oluşturmak ve **raporlama sorgularını read replica'ya taşımak**.
- B. Bir **read replica** oluşturmak ve sipariş uygulamasını **primary DB instance ile read replica arasında dağıtmak**.
- C. Sipariş uygulamasını **on-demand capacity** kullanan **Amazon DynamoDB**'ye taşımak.
- D. Raporlama sorgularını **yoğun olmayan saatlere** (non-peak hours) planlamak.

#### Soru Analizi:

##### Sorunun Analizi

##### Mevcut Durum

- **Amazon RDS for MySQL**
- Aynı veritabanı üzerinde:
  - **Order processing (kritik, write + read)**
  - **Uzun süren reporting sorguları (read-heavy)**

##### Problem

- Reporting sorguları:
  - CPU / I/O tüketiyor
  - Primary DB'yi meşgul ediyor
- Sonuç:
  - **Order processing timeout**

##### Gereksinimler

1. **Timeout'ları ortadan kaldırmak**
2. **Raporlama sorgularını engellememek**
3. **Minimum mimari değişiklik**

 Bu tip senaryo:

## **OLTP (transaction) + OLAP (reporting) çakışması**

### **AWS Tasarım Mantığı**

**İş Yükü**                    **En Uygun Çözüm**

Transactional (sipariş) Primary DB

Read-heavy reporting **Read replica**

Min değişiklik                Native RDS özelliği

### **Seçenek Analizi:**

#### **A Seçeneği Analizi (DOĞRU)**

**Read replica oluşturmak ve reporting sorgularını read replica'ya taşımak**

#### **Neden doğru?**

- Reporting sorguları **read-only**
- Read replica:
  - Read workload'u primary'den ayırrır
  - Order processing rahatlar
- Yazma yok → data tutarlılığı sorun olmaz
- RDS native çözüm → düşük operasyonel yük

#### **Sınav refleksi:**

*RDS + long-running read queries → Read replica*

#### **B. Ordering app'i primary + replica arasında dağıtmak**

- Read replica:
  - Write kabul etmez
- Uygulama karmaşıklasır
- Yanlış kullanım

#### **C. DynamoDB'ye migration**

- Büyük mimari değişiklik
- Relational sorgular kaybolur
- Gereksiz ve riskli

## D. Reporting'i non-peak hours'a taşımak

- İş gereksinimine ters
- Sorunu çözmez, erteler
- Otomatik/teknik çözüm değil

## Sonuç

## Sınav İpuçları

Soruda şu ifadeleri görürsen:

- **Timeout**
- **Long-running reporting queries**
- **RDS**
- **Do not block reporting**

## Read replica

### Cevap (çok kısa):

- **Problem:** Uzun süren raporlama sorguları, RDS primary DB'yi yavaşlatıyor ve siparişlerde timeout oluyor.
- **Çözüm:** Okuma yükünü ayırmak gereklidir.
- **Doğru seçenek:** A — **Read replica oluştur ve raporlama sorgularını read replica'ya taşı.**

### Ezber:

*RDS + uzun süren read sorguları + timeout = Read replica*

---

## QUESTION 62

A hospital wants to create digital copies for its large collection of historical written records. The hospital will continue to add hundreds of new documents each day. The hospital's data team will scan the documents and will upload the documents to the AWS Cloud.

A solutions architect must implement a solution to analyze the documents, extract the medical information, and store the documents so that an application can run SQL queries on the data. The solution must maximize scalability and operational efficiency.

Which combination of steps should the solutions architect take to meet these requirements? (Choose two.)

- A. Write the document information to an Amazon EC2 instance that runs a MySQL database.
- B. Write the document information to an Amazon S3 bucket. Use Amazon Athena to query the data.
- C. Create an Auto Scaling group of Amazon EC2 instances to run a custom application that processes the scanned files and extracts the medical information.
- D. Create an AWS Lambda function that runs when new documents are uploaded. Use Amazon Rekognition to convert the documents to raw text. Use Amazon Transcribe Medical to detect and extract relevant medical information from the text.
- E. Create an AWS Lambda function that runs when new documents are uploaded. Use Amazon Textract to convert the documents to raw text. Use Amazon Comprehend Medical to detect and extract relevant medical information from the text.

**Soru:**

Bir hastane, **büyük bir tarihi yazılı kayıt koleksiyonu** için **dijital kopyalar** oluşturmak istemektedir. Hastane, her gün **yüzlerce yeni belge** eklemeye devam edecektir. Hastanenin **veri ekibi**, belgeleri tarayacak ve belgeleri **AWS Cloud'a** yükleyecektir.

Bir solutions architect'in; belgeleri **analiz eden, tıbbi bilgileri çıkaran** ve bir uygulamanın veriler üzerinde **SQL sorguları çalıştırılabilmesi** için belgeleri saklayan bir çözüm uygulaması gerekmektedir. Çözüm, **ölçeklenebilirliği** ve **operasyonel verimliliği en üst düzeye çıkarmalıdır.**

Bu gereksinimleri karşılamak için solutions architect aşağıdaki adımlardan hangilerini seçmelidir? (**İki tane seçin.**)

- A. Belge bilgilerini, **MySQL veritabanı** çalıştırılan bir **Amazon EC2 instance**'ına yazmak.
- B. Belge bilgilerini bir **Amazon S3 bucket**'a yazmak ve verileri sorgulamak için **Amazon Athena** kullanmak.
- C. Taranan dosyaları işleyen ve tıbbi bilgileri çıkaran özel bir uygulama çalışırmak için **Amazon EC2 instance'larından oluşan bir Auto Scaling group** oluşturmak.
- D. Yeni belgeler yüklenliğinde çalışan bir **AWS Lambda fonksiyonu** oluşturmak. Belgeleri ham metne dönüştürmek için **Amazon Rekognition** kullanmak. Metinden ilgili tıbbi bilgileri tespit etmek ve çıkarmak için **Amazon Transcribe Medical** kullanmak.
- E. Yeni belgeler yüklenliğinde çalışan bir **AWS Lambda fonksiyonu** oluşturmak. Belgeleri ham metne dönüştürmek için **Amazon Textract** kullanmak. Metinden ilgili tıbbi bilgileri tespit etmek ve çıkarmak için **Amazon Comprehend Medical** kullanmak.

**Soru Analizi:**



## 📌 Temel Gereksinimler

1. **Yazılı (taralı) belgeler** → OCR gereklidir
2. **Tıbbi bilgi çıkarımı** (medical entities)
3. **Her gün yüzlerce belge** → yüksek ölçülebilirlik
4. **SQL ile sorgulanabilir veri**
5. **Minimum operasyonel yük**

👉 Bu; **serverless + managed AI/ML + data lake** mimarisi gerektirir.

## 🔍 Doğru AWS Servis Eşleşmesi

İhtiyaç	En Uygun Servis
Belgeleri saklama	<b>Amazon S3</b>
OCR (taralı belge → metin)	<b>Amazon Textract</b>
Tıbbi bilgi çıkarımı	<b>Amazon Comprehend Medical</b>
Olay tetikleme	<b>AWS Lambda (S3 event)</b>
SQL sorguları	<b>Amazon Athena</b>
Düşük ops	<b>Serverless &amp; managed</b>

### Seçenek Analizi:

**B Seçeneği Analizi (DOĞRU)**

**Verileri Amazon S3'e yazmak ve Amazon Athena ile sorgulamak**

### ✓ Neden doğru?

- S3: sınırsız ölçek, düşük maliyet
- Athena: **serverless SQL**, altyapı yok
- Data lake mimarisi
- Operasyonel yük minimum

**E Seçeneği Analizi (DOĞRU)**

**Lambda + Textract + Comprehend Medical**

### ✓ Neden doğru?

- **Textract**: taralı belgeler için **OCR** (Rekognition değil)

- **Comprehend Medical:** tıbbi varlıklarını (ilaç, teşhis, prosedür) çıkarır
- **Lambda:** S3 upload ile otomatik tetiklenir
- Yüksek ölçülebilirlik, yönetim yok

📌 Sınav refleksi:

*Scanned documents → Textract*

*Medical NLP → Comprehend Medical*

### ✗ Diğer Seçeneklerin Analizi

#### ✗ A. EC2 + MySQL

- Ölçülebilirlik sınırlı
- Sunucu/DB yönetimi gereklidir
- SQL var ama **ops yükü yüksek**

#### ✗ C. EC2 Auto Scaling + custom app

- Custom OCR/NLP geliştirme
- Yüksek bakım ve operasyon
- Managed değil

#### ✗ D. Rekognition + Transcribe Medical

- **Rekognition:** görüntü/video analizi, OCR için uygun değil
- **Transcribe Medical:** ses → metin (audio), yazılı belge değil
- Servisler yanlış eşleşmiş

🎯 Sonuç

#### 💡 Sınav İpuçları (Altın Kurallar)

- **Scanned document OCR → Amazon Textract**
- **Medical text extraction → Comprehend Medical**
- **SQL on S3 → Athena**
- **Low ops / scalable → Serverless**

Cevap (en kısa haliyle):

- **Doğru seçenekler: B ve E**

Neden?

- **B:** Belgeleri Amazon S3'te saklayıp Amazon Athena ile serverless SQL sorguları çalıştırırırsın.
- **E: AWS Lambda** ile otomatik tetikleme, **Amazon Textract** ile **OCR**, **Amazon Comprehend Medical** ile **tıbbi bilgi çıkarımı** yapılır.

**Ezber:**

*Taralı tıbbi belgeler + SQL + düşük operasyonel yük = S3 + Athena + Textract + Comprehend Medical*

---

### QUESTION 63

A company is running a batch application on Amazon EC2 instances. The application consists of a backend with multiple Amazon RDS databases. The application is causing a high number of reads on the databases. A solutions architect must reduce the number of database reads while ensuring high availability.

What should the solutions architect do to meet this requirement?

- A. Add Amazon RDS read replicas.
- B. Use Amazon ElastiCache for Redis.
- C. Use Amazon Route 53 DNS caching
- D. Use Amazon ElastiCache for Memcached.

**Soru:**

Bir şirket, Amazon EC2 instance'ları üzerinde çalışan bir batch (toplu işleme) uygulaması kullanmaktadır. Uygulama, birden fazla Amazon RDS veritabanından oluşan bir backend'e sahiptir. Uygulama, veritabanları üzerinde çok sayıda okuma (read) işlemi oluşturmaktadır. Bir solutions architect, yüksek erişilebilirliği (high availability) korurken veritabanı okuma sayısını azaltmalıdır.

Bu gereksinimi karşılamak için solutions architect ne yapmalıdır?

- A. Amazon RDS read replica'ları eklemek.
- B. Amazon ElastiCache for Redis kullanmak.
- C. Amazon Route 53 DNS caching kullanmak.
- D. Amazon ElastiCache for Memcached kullanmak.

**Soru Analizi:**

#### ☒ Mevcut durum

- Uygulama: Amazon EC2 üzerinde çalışan **batch application**

- Backend: **Birden fazla Amazon RDS veritabanı**
- Problem: **Çok fazla read (okuma) işlemi** → veritabanı yükü artıyor
- Gereksinimler:
  - Veritabanı okuma sayısını **azaltmak**
  - **High availability (yüksek erişilebilirlik)** sağlamak

## ?

### Önemli ipuçları

- Okuma ağırlıklı bir iş yükü var
- Batch application → veriler genellikle **tekrar tekrar okunur**
- “Reduce database reads” ifadesi → **cache** güçlü bir aday
- High availability → tek noktaya bağımlı çözümler uygun değil

### Seçenek Analizi:

#### B. Amazon ElastiCache for Redis kullanmak

##### Ne sağlar?

- In-memory cache → **çok hızlı okuma**
- Veriler RDS'den okunur, Redis'te tutulur
- Redis:
  - Replication
  - Multi-AZ
  - Automatic failover destekler

##### Avantajlar:

- RDS'ye yapılan read sayısı ciddi biçimde azalır
- Yüksek performans + yüksek erişilebilirlik

##### Değerlendirme:

- ✓ Database reads ciddi şekilde azalır
- ✓ High availability desteklenir
- ✓ Batch ve read-heavy workload'lar için ideal

#### A. Amazon RDS read replica'ları eklemek

##### Ne sağlar?

- Okuma trafiğini birden fazla read replica'ya dağıtır

- Primary DB üzerindeki okuma yükünü azaltır
- Read replica'lar farklı AZ'lerde olabilir → high availability

**Ancak:**

- Okumalar hâlâ **RDS'ye gider**
- Her sorgu yine veritabanı erişimi gerektirir
- Cache kadar etkili değildir

**Değerlendirme:**

- ✓ Okuma yükünü dağıtır  
✗ Veritabanı okuma sayısını kökten azaltmaz

**✗ C. Amazon Route 53 DNS caching kullanmak**

**Ne sağlar?**

- DNS çözümlemelerini cache'ler
- IP adresi sorgularını hızlandırır

**Ama:**

- Veritabanı okuma yüküyle **hiçbir ilgisi yok**
- Uygulama verisini cache'lemez

**Değerlendirme:**

- ✗ Sorunla tamamen alakasız

**✗ D. Amazon ElastiCache for Memcached kullanmak**

**Ne sağlar?**

- In-memory cache
- Veritabanı okuma sayısını azaltır

**Ancak:**

- Memcached:
  - Persistence yok
  - Replication yok
  - Failover yok

**Dezavantaj:**

- High availability gereksinimini tam karşılamaz

### **Değerlendirme:**

- ✓ Read sayısını azaltır
- ✗ High availability zayıf

### **Genel Karşılaştırma Tablosu**

<b>Seçenek</b>	<b>DB Read Azaltma High Availability Uygunluk</b>		
A (Read Replica)	Orta	Var	✗
B (ElastiCache Redis)	Çok yüksek	Var	✓
C (Route 53)	Yok	Yok	✗
D (Memcached)	Yüksek	Zayıf	✗

### **Sonuç**

### **B. Use Amazon ElastiCache for Redis**

#### **Neden?**

- Veritabanı okuma sayısını en etkili şekilde azaltır
- In-memory cache ile yüksek performans sağlar
- Replication ve failover ile **high availability** sunar
- AWS sınav senaryolarında read-heavy + HA için **en doğru çözüm**dur

---

### **QUESTION 64**

A company needs to run a critical application on AWS. The company needs to use Amazon EC2 for the application's database. The database must be highly available and must fail over automatically if a disruptive event occurs.

Which solution will meet these requirements?

- A. Launch two EC2 instances, each in a different Availability Zone in the same AWS Region. Install the database on both EC2 instances. Configure the EC2 instances as a cluster. Set up database replication.
- B. Launch an EC2 instance in an Availability Zone. Install the database on the EC2 instance. Use an Amazon Machine Image (AMI) to back up the data. Use AWS CloudFormation to automate provisioning of the EC2 instance if a disruptive event occurs.

C. Launch two EC2 instances, each in a different AWS Region. Install the database on both EC2 instances. Set up database replication. Fail over the database to a second Region.

D. Launch an EC2 instance in an Availability Zone. Install the database on the EC2 instance. Use an Amazon Machine Image (AMI) to back up the data. Use EC2 automatic recovery to recover the instance if a disruptive event occurs.

**Soru:**

Bir şirket AWS üzerinde kritik bir uygulama çalıştırılmak istemektedir. Şirket, uygulamanın veritabanı için Amazon EC2 kullanmak zorundadır. Veritabanı **yüksek erişilebilir** olmalı ve **yıkıcı (disruptive)** bir olay gerçekleştiğinde **otomatik olarak failover** yapabilmelidir.

Bu gereksinimleri **hangi çözüm karşıları?**

A. Aynı AWS Region içinde, farklı Availability Zone'larda iki adet EC2 instance başlatın. Veritabanını her iki EC2 instance üzerine kurun. EC2 instance'ları bir cluster olarak yapılandırın. Veritabanı replikasyonunu ayarlayın.

B. Bir Availability Zone içinde bir EC2 instance başlatın. Veritabanını EC2 instance üzerine kurun. Veriyi yedeklemek için bir Amazon Machine Image (AMI) kullanın. Yıkıcı bir olay gerçekleştiğinde EC2 instance'ı otomatik olarak yeniden oluşturmak için AWS CloudFormation kullanın.

C. Farklı AWS Region'larda iki adet EC2 instance başlatın. Veritabanını her iki EC2 instance üzerine kurun. Veritabanı replikasyonunu ayarlayın. Veritabanı failover işlemini ikinci Region'a yapın.

D. Bir Availability Zone içinde bir EC2 instance başlatın. Veritabanını EC2 instance üzerine kurun. Veriyi yedeklemek için bir Amazon Machine Image (AMI) kullanın. Yıkıcı bir olay gerçekleştiğinde instance'ı kurtarmak için EC2 automatic recovery özelliğini kullanın.

**Soru Analizi:**

Şirketin **zorunlu gereksinimleri**:

1. **Veritabanı EC2 üzerinde çalışmalı**  
→ RDS, Aurora gibi managed servisler **kullanılamaz**.
2. **High Availability (Yüksek Erişilebilirlik)**  
→ Tek instance / tek AZ yeterli **değil**.
3. **Otomatik failover**  
→ Manuel müdahale gerektiren çözümler elenir.

4. **Disruptive event** (AZ failure gibi ciddi kesintiler)

→ Aynı AZ içindeki çözümler risklidir.

**Seçenek Analizi:**

A. Aynı Region, farklı AZ'lerde 2 EC2 + DB cluster + replikasyon

**Artıları**

- Multi-AZ mimari → yüksek erişilebilirlik
- AZ failure durumunda diğer AZ çalışır
- DB replikasyonu ile **otomatik failover mümkün**dir

**Eksileri**

- Yönetimsel olarak karmaşık ama **gereksinimleri karşılıyor**

**Gereksinimleri TAM** karşılar

B. Tek EC2 + AMI + CloudFormation

**Yanlış**

- Tek AZ → **high availability yok**
- AMI + CloudFormation → **disaster recovery, failover değil**
- Otomatik olsa bile **kesinti süresi uzun**

Elenir

C. Farklı Region'larda EC2 + replikasyon

**Yanlış**

- Cross-Region DB failover **genelde manuel**
- Daha pahalı ve karmaşık
- Soru **bunu özellikle istemiyor**

Overkill + otomatik failover garanti değil

D. Tek EC2 + automatic recovery

**Yanlış**

- Automatic recovery → **aynı AZ içinde**
- AZ failure durumunda çalışmaz

- High availability sağlamaz

 Elenir

 Sonuç

#### **Kısa Özeti (Sınav Mantığı)**

- EC2 üzerinde veritabanı şart → Managed DB servisleri (RDS/Aurora) yok
- High availability gereklidir → Tek AZ / tek instance olmaz
- Otomatik failover isteniyor → Backup, AMI, recovery yeterli değil

Bu üç koşulu aynı anda karşılayan tek seçenek:

A.

- Farklı Availability Zone'larda iki EC2
- Veritabanı cluster + replikasyon
- AZ failure durumunda otomatik failover

Ezber cümlesi (sınav taktiği):

*EC2 üzerinde veritabanı + high availability + otomatik failover = Multi-AZ EC2 cluster + DB replication*

---

## QUESTION 65

A company's order system sends requests from clients to Amazon EC2 instances. The EC2 instances process the orders and then store the orders in a database on Amazon RDS. Users report that they must reprocess orders when the system fails. The company wants a resilient solution that can process orders automatically if a system outage occurs.

What should a solutions architect do to meet these requirements?

- Move the EC2 instances into an Auto Scaling group. Create an Amazon EventBridge (Amazon CloudWatch Events) rule to target an Amazon Elastic Container Service (Amazon ECS) task.
- Move the EC2 instances into an Auto Scaling group behind an Application Load Balancer (ALB). Update the order system to send messages to the ALB endpoint.
- Move the EC2 instances into an Auto Scaling group. Configure the order system to send messages to an Amazon Simple Queue Service (Amazon SQS) queue. Configure the EC2 instances to consume messages from the queue.

D. Create an Amazon Simple Notification Service (Amazon SNS) topic. Create an AWS Lambda function, and subscribe the function to the SNS topic. Configure the order system to send messages to the SNS topic. Send a command to the EC2 instances to process the messages by using AWS Systems Manager Run Command.

**Soru:**

Bir şirketin sipariş sistemi, istemcilerden gelen istekleri Amazon EC2 instance'larına göndermektedir. EC2 instance'ları siparişleri işler ve ardından siparişleri Amazon RDS üzerindeki bir veritabanında saklar. Kullanıcılar, sistem başarısız olduğunda siparişleri yeniden işlemek zorunda kaldıklarını bildirmektedir. Şirket, bir sistem kesintisi (outage) meydana geldiğinde siparişlerin **otomatik olarak işlenmeye devam edebileceği, dayanıklı (resilient)** bir çözüm istemektedir.

Bu gereksinimleri karşılamak için bir solutions architect ne yapmalıdır?

**A.** EC2 instance'larını bir Auto Scaling grubuna taşıyın. Bir Amazon EventBridge (Amazon CloudWatch Events) kuralı oluşturun ve bu kuralı bir Amazon Elastic Container Service (Amazon ECS) task'ini hedefleyecek şekilde yapılandırın.

**B.** EC2 instance'larını bir Application Load Balancer (ALB) arkasında çalışan bir Auto Scaling grubuna taşıyın. Sipariş sistemini, mesajları ALB endpoint'ine gönderecek şekilde güncelleyin.

**C.** EC2 instance'larını bir Auto Scaling grubuna taşıyın. Sipariş sistemini, mesajları bir Amazon Simple Queue Service (Amazon SQS) kuyruğuna gönderecek şekilde yapılandırın. EC2 instance'larını, mesajları bu kuyrukta tüketecek (consume edecek) şekilde yapılandırın.

**D.** Bir Amazon Simple Notification Service (Amazon SNS) topic'ı oluşturun. Bir AWS Lambda fonksiyonu oluşturun ve bu fonksiyonu SNS topic'ine abone edin. Sipariş sistemini, mesajları SNS topic'ine gönderecek şekilde yapılandırın. AWS Systems Manager Run Command kullanarak EC2 instance'larına mesajları işleme komutu gönderin.

**Soru Analizi:**

**Temel problemler:**

- Sistem **fail olduğunda siparişler kayboluyor**
- Kullanıcılar siparişleri **yeniden göndermek zorunda kalıyor**
- Bu durum **dayanıksız (non-resilient)** bir mimari olduğunu gösterir

**İstenenler:**

1. **Resilient (dayanıklı) çözüm**

2. Outage sırasında siparişlerin kaybolmaması
3. Sistem geri geldiğinde siparişlerin otomatik işlenmesi
4. Minimum manuel müdahale

👉 Bu tip senaryolarda AWS'nin klasik çözümü:

**Asenkron mimari + mesaj kuyruğu (queue)**

**Seçenek Analizi:**

**✓ C. ASG + Amazon SQS (Queue)**

- SQS → **dayanıklı, kalıcı mesaj kuyruğu**
- EC2'ler kapalı olsa bile:
  - Siparişler **kuyrukta saklanır**
  - Sistem geri geldiğinde **otomatik işlenir**
- Auto Scaling ile tüketici sayısı artırılabilir
- Loose coupling (gevşek bağlı mimari)

**✓ AWS'in önerdiği altın standart çözüm**

**✗ A. ASG + EventBridge + ECS task**

**✗ Yanlış**

- EventBridge olay bazlı tetikleme içindir
- Siparişleri **buffer'layamaz**
- Outage sırasında mesajlar **kalıcı olarak tutulmaz**

**✗ Dayanıklılık sağlamaz**

**✗ B. ASG + ALB**

**✗ Yanlış**

- Load Balancer sadece trafiği dağıtır
- EC2'ler kapalıysa istekler **kaybolur**
- **Retry / buffering yok**

**✗ Resilient değil**

**✗ D. SNS + Lambda + SSM Run Command**

## Yanlış

- SNS **push** tabanlıdır, mesajlar kısa ömürlüdür
- Lambda + SSM Run Command → gereksiz karmaşıklık
- Outage sırasında mesajlar **kaçabilir**

 Operasyonel yük yüksek, resiliency düşük

## Sonuç

### Kısa ve Net Gerekçe

- Sistem kesintilerinde siparişler **kaybolmamalı**
- Siparişler **dayanıklı bir şekilde saklanmalı**
- Sistem geri geldiğinde siparişler **otomatik işlenmeli**

 **Amazon SQS**, bu gereksinimler için tasarlanmış bir **mesaj kuyruğu** servisidir:

- Mesajları **kalıcı olarak saklar**
- Tüketiciler (EC2'ler) kapalı olsa bile veri kaybı olmaz
- Auto Scaling ile birlikte çalışır
- Loose coupling sağlar

## Sınav Ezberi

**Resilient order processing = SQS + Auto Scaling**

**Outage varsa → Queue kullan**

---