





웹 트래픽이 지리적으로 분산된 경우, 전체 인프라를 전 세계에 복제하는 것은 때때로 불가능할 수도 있습니다(그리고 꼭 비용 효율적이지만은 않습니다). CDN에서는 웹 콘텐츠(예: 비디오, 웹 페이지, 이미지 등)의 캐시된 사본을 고객에게 제공하기 위해 엣지 로케이션의 글로벌 네트워크를 사용할 수 있습니다. 응답 시간을 줄이기 위해 CDN은 고객 또는 발신 요청 위치에 가장 가까운 엣지 로케이션을 사용합니다. 웹 자산이 캐시에서 제공되면 처리량은 크게 증가합니다. 동적 데이터의 경우, 오리진 서버에서 데이터를 검색하도록 많은 CDN을 구성할 수 있습니다.

## Amazon CloudFront



Amazon의 글로벌 CDN(콘텐츠 전송 네트워크)

기본적인 멀티 티어 캐시와 광범위한 유연성으로 모든 전송 사례에 최적화

아키텍처에 추가적인 보안 계층 제공

WebSocket 프로토콜 지원

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Amazon CloudFront는 웹 사이트, API, 동영상 콘텐츠 또는 기타 웹 자산의 전송을 가속화하는 글로벌 CDN 서비스입니다. 다른 AWS 제품과 통합하여 사용하면 개발자와 기업이 최소 사용 약정 없이도 최종 사용자에게 쉽고 빠르게 콘텐츠를 전송할 수 있습니다.

Amazon CloudFront는 지연 시간 및 처리량을 위한 네트워크 계층 최적화와 함께 캐시 동작 최적화를 위한 강력한 유연성을 제공하는 데 최적화되어 있습니다.

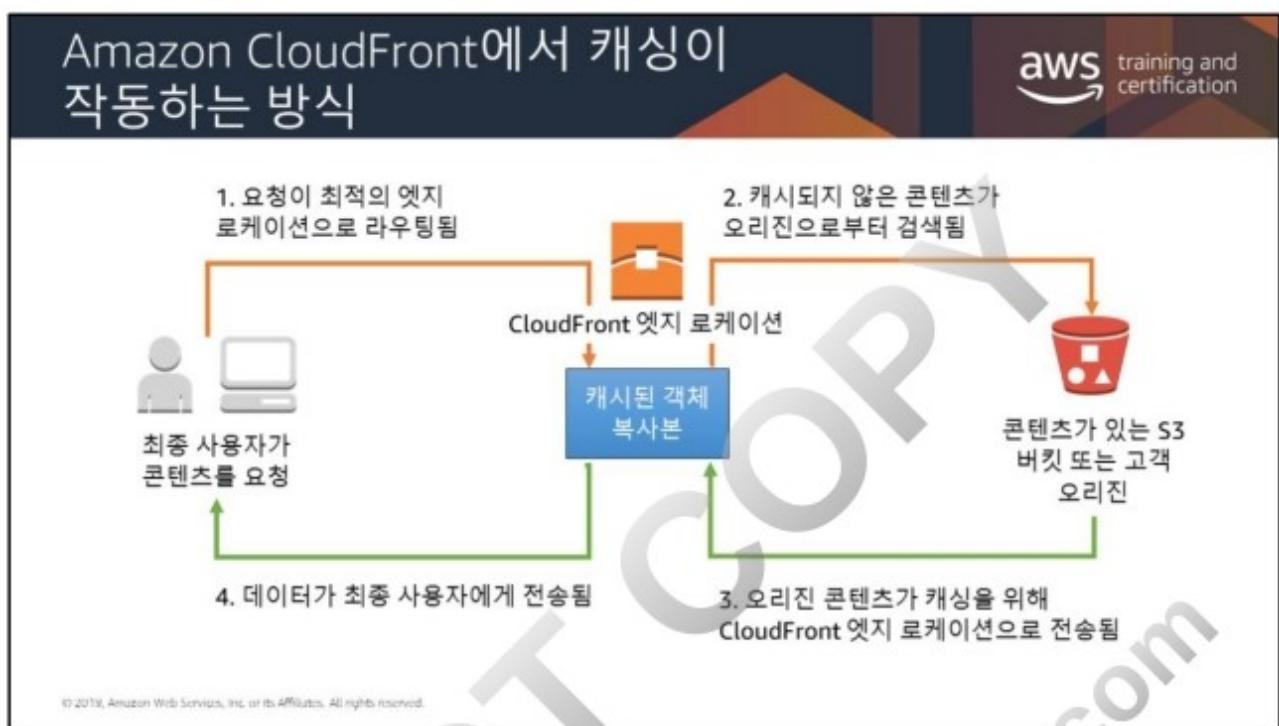
콘텐츠 전송 네트워크(CDN)는 멀티 티어 캐시를 기본적으로 제공하며, 이와 더불어 객체가 아직 엣지에 캐시되지 않았을 때 오리진 서버의 지연 시간을 개선하고 부하를 줄여주는 리전 엣지 캐시도 함께 제공합니다.

CloudFront는 짧은 지연 시간과 빠른 데이터 전송 속도로 콘텐츠를 쉽게 배포할 수 있는 비용 효율적인 방법을 제공합니다. 모든 AWS 인프라 서비스와 마찬가지로 CloudFront는 장기 약정 또는 최소 요금이 필요하지 않은 종량 과금제 서비스입니다. CloudFront를 사용하면 엣지 로케이션의 글로벌 네트워크를 사용해 최종 사용자에게 파일이 전송됩니다.

Amazon CloudFront는 WebSocket 프로토콜을 통한 실시간 양방향 통신을 지원합니다. 이 영구 연결을 통해 클라이언트와 서버가 반복적인 연결로 인한 오버헤드 없이 서로 실시간 데이터를 전송할 수 있습니다. 이는 특히 채팅, 공동 작업, 게임, 금융 거래 같은 통신 애플리케이션에 유용합니다.

CloudFront의 WebSocket 지원으로 고객은 다른 동적 및 정적 콘텐츠와 동일한 경로를 통해 WebSocket 트래픽을 관리할 수 있습니다. CloudFront의 엣지 로케이션 글로벌 네트워크로 트래픽이 사용자에 더 가까워지고, 응답성 및 안정성이 개선되며, 고객은 AWS Shield 및 AWS WAF에 기본 통합된 CloudFront를 통한 DDoS 보호를 이용할 수 있습니다.

WebSocket 프로토콜은 클라이언트가 업데이트된 데이터를 수신하거나 새로운 정보를 보내려 할 때마다 계속 새 연결을 열 필요가 없기 때문에 일반적 TCP 연결의 오버헤드를 제거합니다. WebSocket 연결은 클라이언트 또는 서버에서 닫을 때까지 계속 열려 있으며, 연결이 열려 있는 동안 데이터를 주고받을 수 있습니다.



CloudFront를 통해 서비스하는 콘텐츠를 최종 사용자가 요청하면 자연 시간(시간 지연)이 가장 짧은 엣지 로케이션으로 사용자가 라우팅되므로 가능한 최고의 성능으로 콘텐츠가 제공됩니다. 콘텐츠가 이미 자연 시간이 가장 짧은 엣지 로케이션에 있는 경우, CloudFront가 콘텐츠를 즉시 제공합니다. 콘텐츠가 현재 해당 엣지 로케이션에 없는 경우, CloudFront에서는 콘텐츠의 최종 버전에 대한 원본으로 식별한 Amazon S3 버킷 또는 HTTP 서버(예: 웹 서버)에서 콘텐츠를 검색합니다.

상기의 예에서 콘텐츠가 캐시되지 않은 경우, 요청된 객체는 오리진으로부터 검색됩니다. 사용자가 요청한 데이터를 검색하여 반환하기 위해 단계 1, 2, 3 및 4가 진행됩니다.

콘텐츠가 캐시된 경우, 캐시된 객체 요청은 최적의 엣지 로케이션으로 라우팅되고 캐시된 객체는 단계 1과 4에서처럼 검색됩니다.



CloudFront의 캐싱 및 가속화 기술을 통해 AWS는 정적 이미지에서 사용자 입력 콘텐츠까지 모든 콘텐츠를 제공할 수 있습니다.

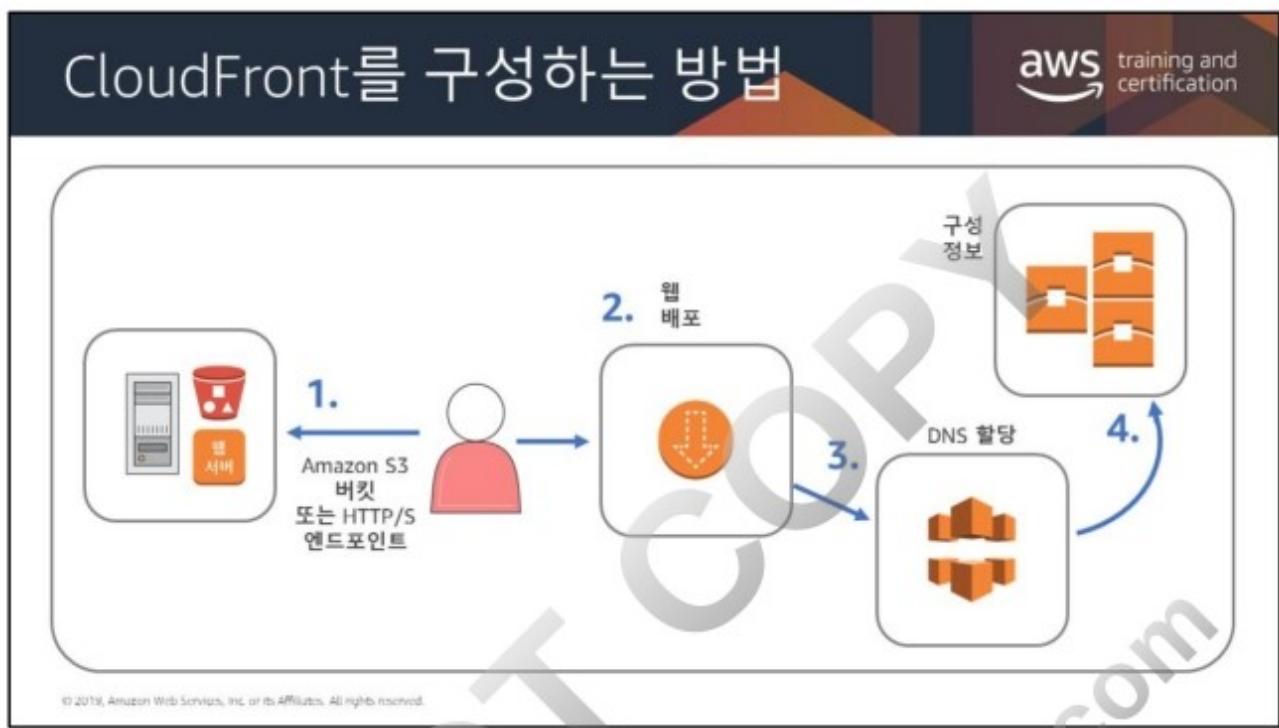
**정적:** TTL (Time-To-Live)이 높은 이미지, js, html 등

**동영상:** rtmp 및 http 스트리밍 지원

**동적:** 사용자 정의 콘텐츠 및 캐싱할 수 없는 콘텐츠

**사용자 입력:** http 작업 지원(Put/Post 등 포함)

**보안:** SSL (HTTPS)을 통해 콘텐츠를 안전하게 제공



1. CloudFront가 사용자의 파일을 가져오는 출처에 해당하는 Amazon S3 버킷 또는 사용자의 HTTP 서버와 같은 오리진 서버를 지정합니다. 이들 서버는 전 세계의 CloudFront 엣지 로케이션에서 배포됩니다.

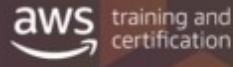
오리진 서버는 객체의 최종 원본 버전을 저장합니다. HTTP를 통해 콘텐츠를 서비스하는 경우, 오리진 서버는 Amazon S3 버킷 또는 HTTP 서버(예: 웹 서버)입니다. HTTP 서버는 Amazon EC2 (Amazon Elastic Compute Cloud) 인스턴스 또는 사용자가 관리하는 온프레미스 서버에서 실행할 수 있습니다. 이들 서버를 일컬어 사용자 지정 오리진이라고도 합니다.

2. 사용자가 웹 사이트나 애플리케이션을 통해 파일을 요청할 경우, 어떤 오리진 서버에서 파일을 가져올지를 CloudFront에 알려주는 CloudFront 배포를 만듭니다. 동시에 CloudFront에서 모든 요청을 기록할지 여부 및 배포를 만들자마자 활성화할지 여부와 같은 세부 사항을 지정합니다.

3. CloudFront는 새 배포에 도메인 이름을 할당합니다.

4. CloudFront는 콘텐츠를 제외한 배포의 구성을 모든 엣지 로케이션에 전달합니다. 엣지 로케이션은 지리적으로 분산된 여러 데이터 센터의 서버들로 이루어진 집합체이며, 여기서 CloudFront는 객체의 사본을 캐시합니다.

## 콘텐츠 만료 방법



Time To Live (TTL)

- 기간 고정(만료 기간)
- 고객이 설정
- CloudFront에서 오리진으로의 GET 요청에 **If-Modified-Since header**를 사용

객체 이름 변경

- Header-v1.jpg를 Header-v2.jpg로 변경
- 새 이름이 생기면 새로 고침이 수행됨

객체 무효화

- 마지막 수단: 매우 비효율적이고 비용이 많이 들

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

캐시된 콘텐츠를 만료하는 방법은 3가지가 있는데, 처음 2가지 방법을 사용하는 것이 좋습니다. 즉시 교체할 필요가 없다면 TTL이 가장 간편합니다. (자세한 내용은

<http://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/RequestAndResponseBehaviorS3Origin.html>을 참조하십시오.)

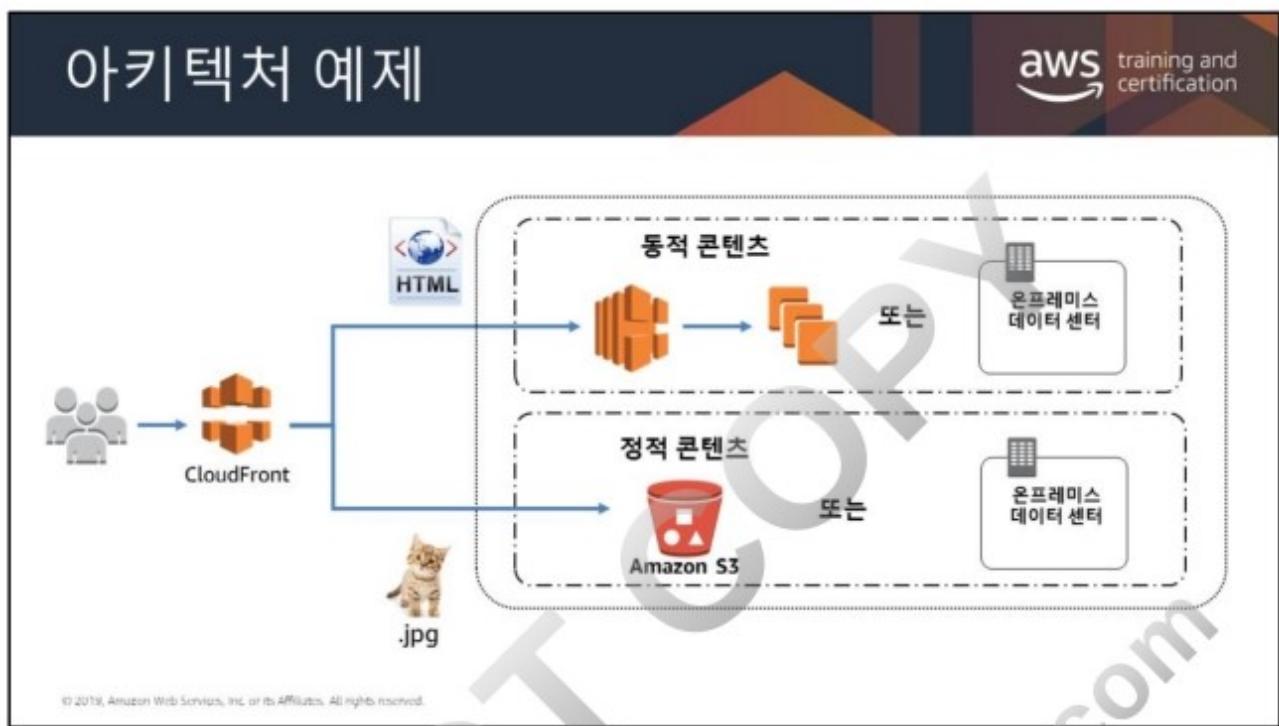
특정 오리진에 대한 TTL을 0으로 설정하더라도 CloudFront는 해당 오리진에서 콘텐츠를 계속 캐시합니다. 그런 다음, If-Modified-Since 헤더를 포함한 GET 요청을 전달함으로써 오리진에서 콘텐츠가 변경되지 않았다면 CloudFront가 캐시된 콘텐츠를 계속 사용할 수 있는지를 오리진이 알려주도록 합니다.

두 번째 방법은 약간의 노력이 더 필요하지만 즉각적입니다(일부 CMS 시스템에서 이 방법을 어느 정도 지원할 수 있음). 고객이 CloudFront 배포에서 기존 객체를 업데이트하고 같은 객체 이름을 사용할 수는 있지만 이는 권장되지는 않습니다. CloudFront는 사용자가 새로운 객체 또는 업데이트된 객체를 오리진에 저장했을 때가 아니라, 해당 객체가 요청되었을 때에만 객체를 엣지 로케이션에 배포합니다. 오리진에 있는 기존 객체를 같은 이름의 최신 버전으로 업데이트하는 경우, 엣지 로케이션은 두 개의 나열된 이벤트가 모두

발생해야 오리진에서 새로운 버전을 가져옵니다.

세 번째 방법은 개별 객체에 대해 매우 드문 경우에만 사용해야 합니다. 이는 결코 좋은 솔루션이 아닙니다(시스템이 모든 엣지 로케이션과 강제로 상호 작용해야 하기 때문).

DO NOT COPY  
zlagusdbs@gmail.com



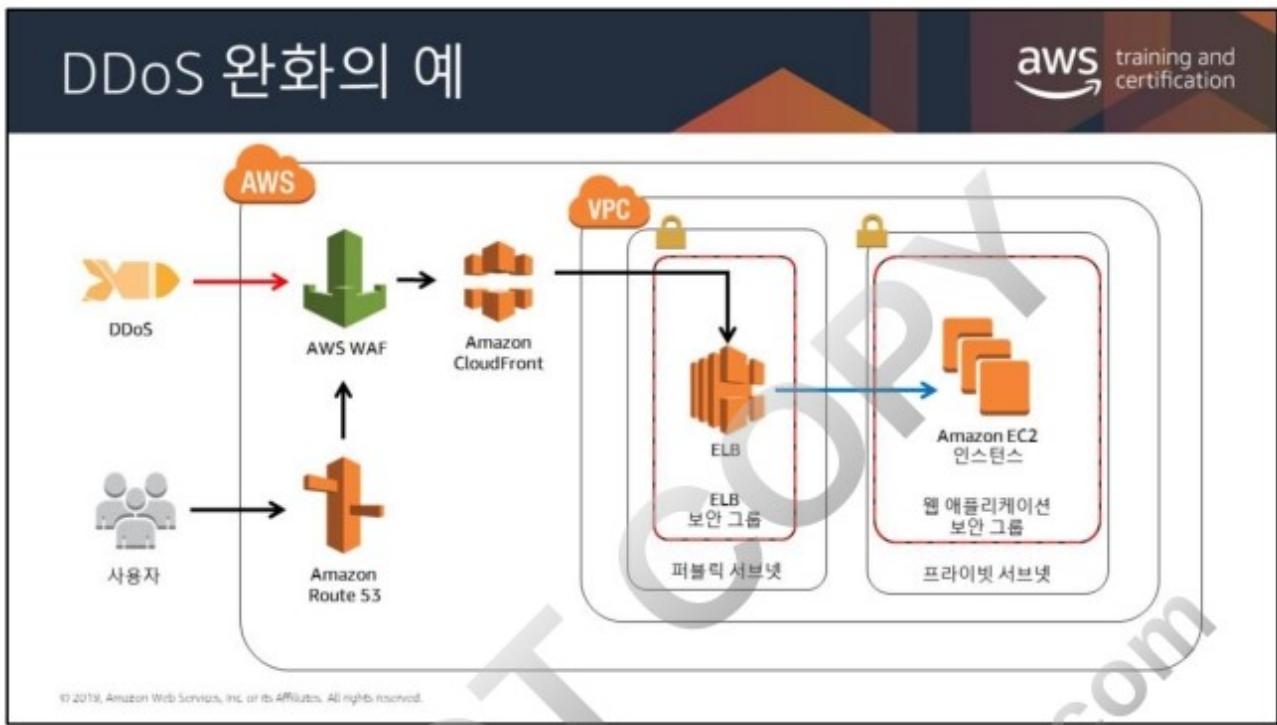
사용자는 대체로 정적 콘텐츠만 캐시합니다. 다만 동적이거나 또는 고유한 콘텐츠는 애플리케이션의 성능에 영향을 미칩니다. 수요에 따라 차이는 있겠지만, Amazon S3에서 동적 콘텐츠 또는 고유한 콘텐츠를 캐싱하면 성능 향상에 도움이 될 수 있습니다.

#### 정적 콘텐츠 오프로딩:

- 정적 자산에 대해 상대 URL 참조 대신, 절대 URL 참조를 생성합니다.
- 정적 자산을 Amazon S3에 저장합니다.
- "WORM (Write Once Read Many)"에 대해 최적화합니다.

또한 콘텐츠를 지리적으로 제한할 수 있습니다. 지리적 차단이라고도 하는 지리적 제한을 사용하면 특정 지리적 위치에 있는 사용자가 CloudFront 웹 배포를 통해 배포한 콘텐츠에 액세스하는 것을 차단할 수 있습니다. 지리적 제한은 다음의 두 가지 방법 중 하나를 선택하여 사용하면 됩니다.

- CloudFront 지리적 제한 기능을 사용하면 배포와 연결된 파일 전체에 대한 액세스를 제한하고 국가 수준에서 액세스를 제한할 수 있습니다.
- 타사 지리적 위치 서비스를 사용하면 배포와 연결된 파일의 하위 집합에 대한 액세스를 제한하거나 국가 수준보다 더 세분화된 범위에서 액세스를 제한할 수 있습니다.



이것은 DDoS 공격을 방지 또는 완화하는 데 도움이 될 수 있는 복원력이 뛰어난 아키텍처의 예에 속합니다.

공격 노출 영역을 최소화하기 위한 전략은 (a) 필요한 인터넷 접속 지점의 수를 축소하고, (b) 중요하지 않은 인터넷 접속 지점을 제거하며, (c) 관리 트래픽에서 최종 사용자 트래픽을 분리하고, (d) 신뢰할 수 없는 최종 사용자가 액세스할 수 없도록 필요한 인터넷 접속 지점을 난독화하며, (e) 공격의 영향을 최소화하기 위해 인터넷 접속 지점의 결합을 해제하는 것입니다. Amazon Virtual Private Cloud (VPC)로 이 전략을 달성할 수 있습니다.

AWS를 사용하면 2가지 형태의 확장, 즉 수평 확장 및 수직 확장을 활용할 수 있습니다. DDoS의 측면에서 볼 때, AWS의 확장을 활용할 수 있는 방법은 3가지가 있습니다. 즉, (1) 사용자의 애플리케이션에 적절한 인스턴스 유형을 선택하고, (2) 자동 확장을 위한 Elastic Load Balancing 및 Auto Scaling 등의 서비스를 구성하며, (3) Amazon CloudFront와 Amazon Route 53과 같은 AWS 글로벌 서비스에 내재화된 확장 기능을 사용하는 것입니다.

ELB는 유효한 TCP 요청만을 지원하기 때문에 UDP 및 SYN 플러드와 같은 DDoS 공격은 인스턴스에 도달할 수 없습니다.

네트워크 트래픽이 높을 때(DDoS 공격의 전형적인 결과) Auto Scaling 그룹에 새 인스턴스를 점진적으로 추가하는 조건을 설정할 수 있습니다.

Elastic Load Balancing 및 Amazon EC2와 같이 AWS 리전에서 제공하는 서비스를 사용하면 DDoS 복원력을 구축할 수 있으며 확장을 통해 특정 리전 내에서 예상치 못한 트래픽 양을 처리할 수 있습니다. Amazon CloudFront, AWS WAF, Amazon Route 53 및 Amazon API Gateway와 같이 AWS 엣지 로케이션에서 제공하는 서비스를 이용하면 애플리케이션에 더 큰 내결함성을 제공하고 더 많은 양의 트래픽을 관리하기 위한 확장성을 증진할 수 있는 엣지 로케이션의 글로벌 네트워크를 활용할 수 있습니다.

이러한 각각의 서비스를 사용하여 인프라 계층 및 애플리케이션 계층에 대한 복원력을 구축하는 데 따른 이점들은 이후 섹션에서 설명하고 있습니다.

Amazon CloudFront에는 유효한 TCP 연결 및 HTTP 요청만 실행하고 유효하지 않은 요청은 폐기할 수 있는 필터링 기능도 포함하고 있습니다.

WAF(웹 애플리케이션 방화벽)는 IP 주소, HTTP 헤더, HTTP 본문 또는 URI 문자열과 같은 데이터를 기반으로 하여 웹 요청을 필터링하기 위해 HTTP 트래픽에 규칙 세트를 적용하는 도구입니다. 불법적인 트래픽을 오프로드하여 DDoS 공격을 완화하는 데 도움을 줄 수 있습니다.

현재 AWS는 관리형 WAF 서비스를 제공하고 있습니다. AWS WAF에 대한 자세한 내용은 <http://docs.aws.amazon.com/waf/latest/developerguide/what-is-aws-waf.html>을 참조하십시오.

백서: AWS Best Practices for DDoS Resiliency:

[https://d0.awsstatic.com/whitepapers/Security/DDoS\\_White\\_Paper.pdf](https://d0.awsstatic.com/whitepapers/Security/DDoS_White_Paper.pdf)





## 세션 관리

aws training and certification

Elastic Load Balancing



고정 세션

사용자 세션을 관리하는 특정 서버로 요청을 라우팅할 수 있습니다

- 클라이언트 측 쿠키
- 비용 효율성
- 세션 검색 속도 증가

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

세션 관리가 캐싱과 관련이 없는 것처럼 보이지 않을 수도 있지만 실제로는 관련이 있습니다.

기본적으로 웹 세션은 동일한 사용자가 환경으로 보내는 일련의 HTTP 트랜잭션입니다. HTTP는 문서를 전송할 수 있도록 설계되었습니다. 이는 상태를 관리하지 않습니다. 모든 요청은 이전 트랜잭션과 무관합니다. 서버에 전달된 모든 요청에 대한 자격 증명을 사용자들이 전송할 필요가 정말로 있다고 생각하십니까? 귀사의 서버는 사용자들과 그들의 요구 사항이 증가함에 따라 확장하는 데 필요한 네트워크 및 컴퓨팅 파워를 보유하고 있습니까?

세션 관리는 인증 및 액세스 제어를 의미합니다. 상태 관리에 대한 일반적인 접근 방식은 고정 세션 또는 분산 캐시의 사용을 포함합니다.

고정 세션(세션 선호도라고도 함)을 사용하면 사용자의 세션을 관리하는 특정 서버로 요청을 라우팅 할 수 있습니다. 세션의 유효성은 로드 밸런서에서 설정된 클라이언트 측 쿠키 또는 파라미터 등 여러 가지 방법으로 결정할 수 있습니다.

사용자는 애플리케이션을 실행하는 웹 서버에 세션을 저장하기 때문에 고정 세션은 비용 효율적입니다. 이는 네트워크 지연 시간을 없애고 해당 세션의 검색 속도를 높입니다. 그러나 장애가 발생할 경우, 하나의 노드에 저장된 여러 세션이 손실될 가능성이 있습니다.

확장 시 활성 세션은 늘어난 용량에 대한 트래픽 라우팅을 차단하기 때문에 트래픽이 여러 서버에 걸쳐 불균일하게 분산될 가능성이 있습니다.

DO NOT COPY  
zlagusdbs@gmail.com

## 상태 정보를 위해 DynamoDB를 사용하는 경우

aws training and certification

사용 사례: 온라인 게임 사이트  
문제: 더 빠른 세션 검색

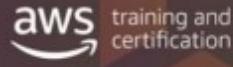
The diagram illustrates a system architecture for an online game site. A user icon sends a request to a CloudFront icon. This request is then processed by a Web Server icon, which finally reaches a Game Server icon. Above this flow, a blue cylinder labeled '세션 상태' (Session State) has a downward arrow pointing to the Web Server icon, indicating that session state information is stored in DynamoDB and retrieved via the web server.

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.





## 데이터베이스 캐싱은 언제 시작해야 합니까?



The slide features three icons illustrating scenarios where database caching might be beneficial:

- Icon 1:** A tablet and a smartphone. **Text:** 고객에 대한 응답 시간이 우려되는 경우 (When response time for customers is a concern).
- Icon 2:** A computer monitor displaying a pie chart. **Text:** 부하가 큰 대용량 요청으로 데이터베이스가 넘치는 것을 알게 되는 경우 (When a large volume of high-pressure requests overburden the database).
- Icon 3:** A briefcase containing a green folder and a blue profile icon. **Text:** 데이터베이스 비용을 줄이고 싶을 때 (When you want to reduce database costs).

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

## 상태 정보를 위해 DynamoDB를 사용하는 경우

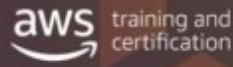
aws training and certification

사용 사례: 온라인 게임 사이트  
문제: 더 빠른 DB 응답 필요

The diagram illustrates a client request flow. A user icon sends a request to a CloudFront icon, which then connects to a Web Server icon. Finally, the request reaches a Game Server icon. Above this flow, a blue cylinder icon represents session state. A red exclamation mark icon is positioned next to the cylinder, with a blue arrow pointing down to the cylinder, labeled '밀리초 단위의 응답 시간' (Response time in milliseconds).

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

# Amazon DynamoDB Accelerator





Amazon  
DynamoDB  
Accelerator

- 탁월한 성능
- 높은 확장성
- 완전관리형
- DynamoDB와 API 호환
- 보안

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

## 탁월한 성능

DynamoDB는 10밀리초 미만의 일관된 지연 시간을 제공합니다. DynamoDB plus DAX는 읽기 중심의 워크로드에 대한 초당 수백만 건의 요청에서 마이크로초 단위의 응답 시간을 제공합니다.

## 뛰어난 확장성

DAX는 온디맨드 조정 기능이 있습니다. 3노드의 DAX 클러스터로 시작하여 최대 10노드의 클러스터에 이르기까지 필요에 따라 용량을 늘릴 수 있습니다.

## 완전 관리형

DynamoDB와 마찬가지로 DAX는 완전관리형 서비스입니다. DAX는 프로비저닝, 설정 및 구성, 소프트웨어 패치 그리고 조정 작업 중 노드를 통한 데이터 복제 등 다양한 관리 작업을 처리합니다. DAX는 장애 탐지, 장애 복구, 소프트웨어 패치와 같은 일반적인 관리 작업들을 자동으로 실행합니다.

## DynamoDB와 API 호환

DAX는 DynamoDB와 API 호환이 되기 때문에 작동 중인 애플리케이션 코드를 변경할 필요가 없습니다. DAX 클러스터를 프로비저닝하고 DAX 클라이언트 SDK를 사용하여 DAX 클러스터에서 기존 DynamoDB API 호출을 가리키면, DAX가 나머지 모든 작업을 처리합니다.

## 유연성

여러 DynamoDB 테이블에 대해 하나의 DAX 클러스터를 프로비저닝하거나 하나의 DynamoDB 테이블에 대해 여러 DAX 클러스터를 프로비저닝하거나 혹은 앞서 언급한 2가지 방법을 조합하여 프로비저닝할 수 있습니다.

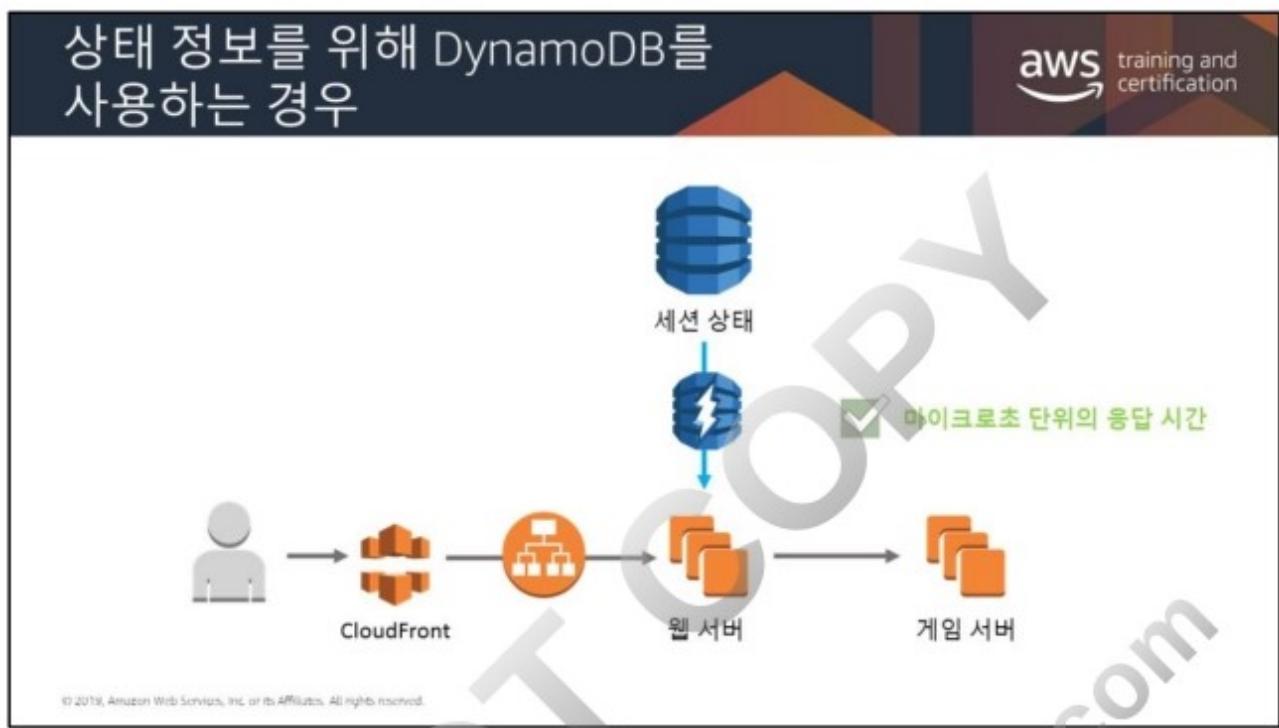
## 보안

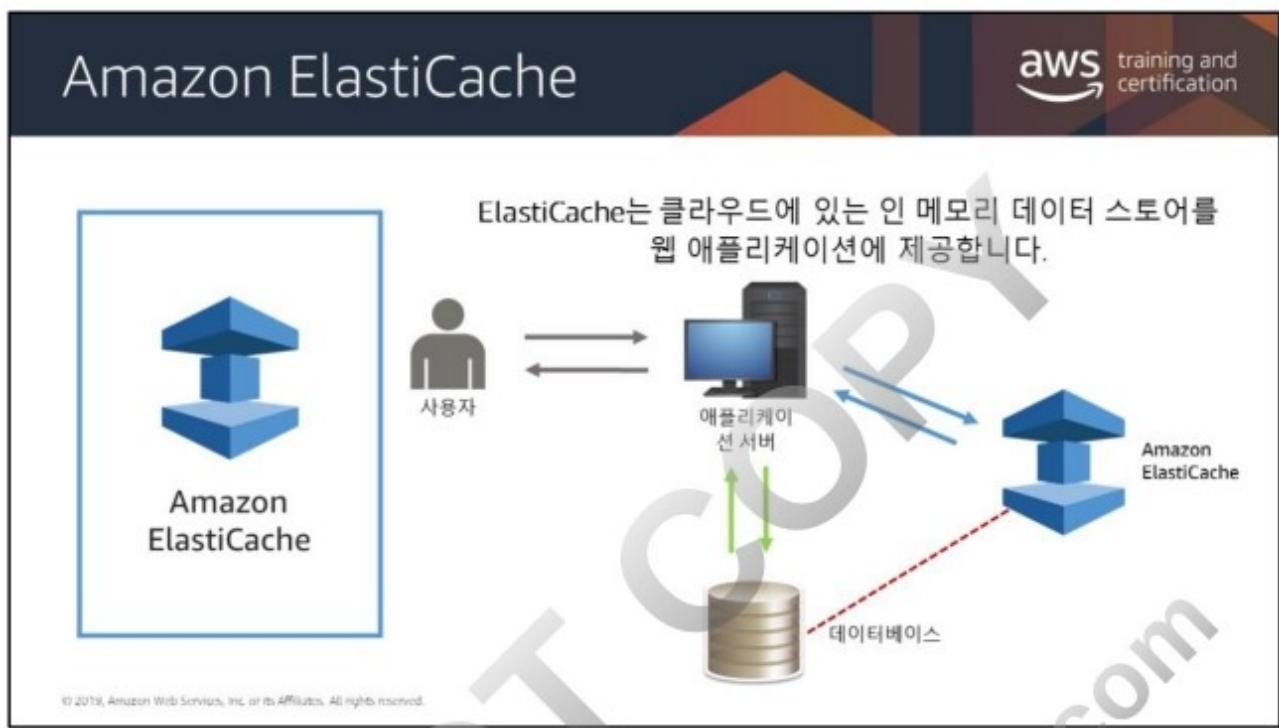
DAX는 AWS 서비스와 완벽하게 통합되어 보안을 강화합니다. AWS Identity and Access Management (IAM)를 사용하면 각 사용자에게 고유한 보안 자격 증명을 할당하고 서비스 및 리소스에 대한 각 사용자의 액세스를 제어할 수 있습니다. Amazon CloudWatch를 사용하면 시스템 전체의 리소스 사용률, 애플리케이션 성능 및 운영 상태를 파악할 수 있습니다. AWS CloudTrail과 통합하면 클러스터 구성의 변경 사항을 손쉽게 기록하고 감사할 수 있습니다. DAX는 기존 애플리케이션에서 안전하고 간편하게 액세스할 수 있도록 Amazon Virtual Private Cloud (VPC)를 지원합니다. 태깅은 DAX 클러스터를 관리하는 데 도움이 되는 가시성을 추가로 제공합니다.

캐시된 데이터를 검색하면 기존 DynamoDB 테이블에서 읽기 부하가 줄어듭니다. 따라서 프로비저닝된 읽기 용량을 줄이면서 전체 운영 비용을 절감할 수 있습니다.

자세한 내용은 다음을 참조하십시오.

<https://aws.amazon.com/blogs/database/amazon-dynamodb-accelerator-dax-a-read-throughwrite-through-cache-for-dynamodb/>





Amazon ElastiCache는 클라우드에서 인 메모리 캐시를 배포, 운영, 조정하는 데 사용되는 웹 서비스입니다. ElastiCache는 비교적 느린 디스크 기반 데이터베이스에 전적으로 의존하기보다는 빠른 관리형 인 메모리 데이터 스토어에서 정보를 검색할 수 있는 기능을 지원함으로써 웹 애플리케이션의 성능을 향상합니다. 가능하다면 애플리케이션은 ElastiCache에서 데이터를 검색하고 캐시에서 데이터를 찾을 수 없을 때에는 데이터베이스를 참조하게 됩니다.

## 어떻게 작동합니까?

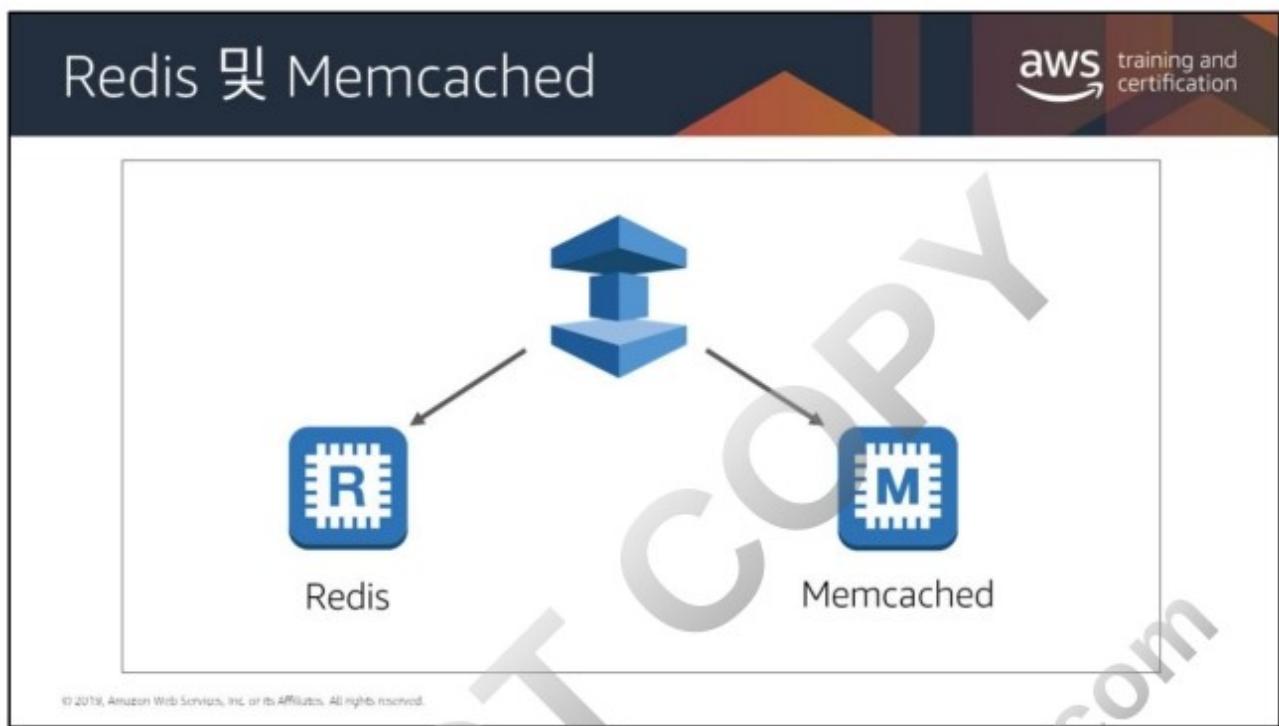
The diagram illustrates the architecture of an Amazon ElastiCache cluster. On the left, a blue 3D cube icon represents 'Amazon ElastiCache'. A line connects it to a rectangular box labeled '클러스터' (Cluster). Inside the cluster box is a 3x5 grid of blue squares, each labeled 'CACHE'. The bottom row of these squares is labeled '캐시 노드' (Cache Node). To the right of the cluster box is a bulleted list:

- 노드는 ElastiCache 배포에서 가장 작은 블록입니다.
- 각 노드에는 고유한 DNS (Domain Name Service) 이름 및 포트가 있습니다
- 완전관리형 서비스

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

캐시 노드는 ElastiCache 배포에서 가장 작은 빌딩 블록입니다. 이것은 다른 노드와 분리되어 존재하거나 혹은 다른 노드와의 일부 관계(클러스터라고도 함)에서도 존재할 수 있습니다.

Amazon ElastiCache는 완전 관리형 서비스이기 때문에 사용하지 않는 캐시 노드를 계속 실행할 필요가 없습니다. 더 많은 용량이 필요할 경우, 그러한 요구를 수용할 수 있도록 클러스터를 확장할 수 있습니다.

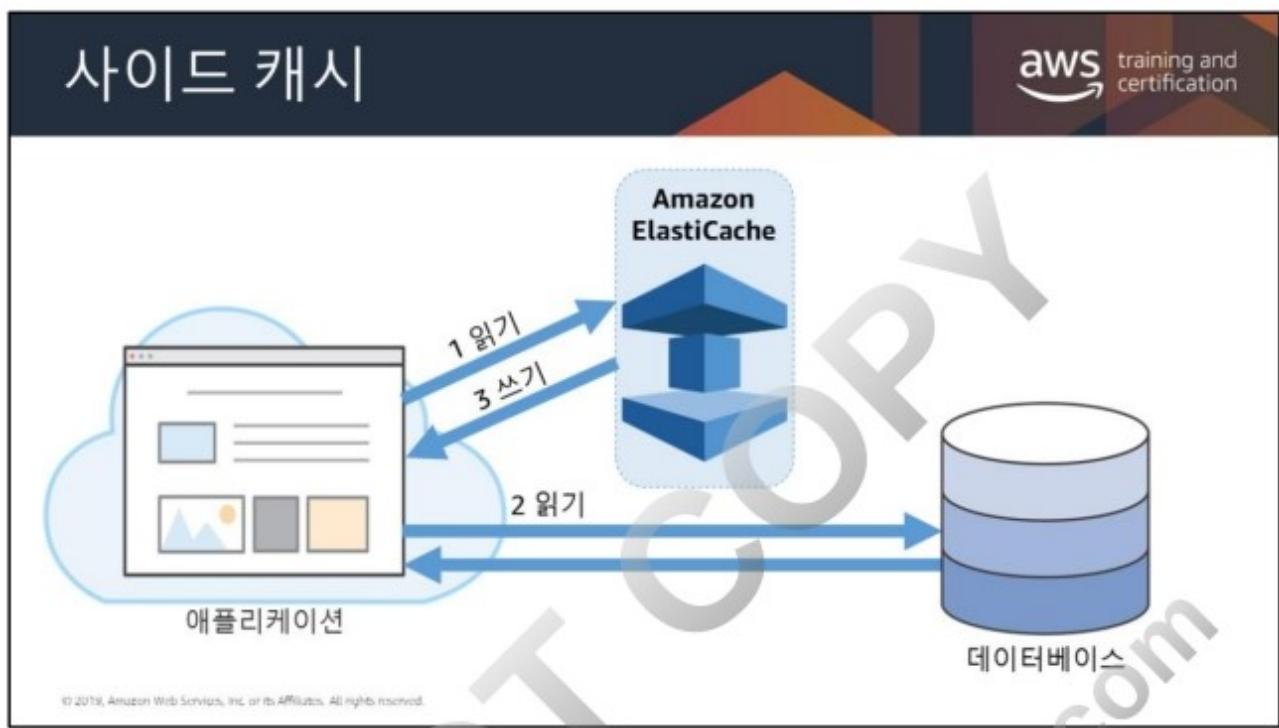


Memcached 사용 ElastiCache는 클러스터당 최대 20개의 노드까지 확장할 수 있으며, Redis용 ElastiCache는 데이터 액세스 성능 향상을 위해 최대 90개의 노드까지 확장할 수 있습니다. ElastiCache는 Amazon VPC를 지원하므로 사용 중인 노드에 대해 선택한 IP 범위로 클러스터를 격리할 수 있습니다.

ElastiCache는 다른 AWS 서비스에서 사용하는 것과 동일한 고안정성 인프라에서 실행됩니다. Redis 워크로드의 경우, ElastiCache는 자동 장애 조치가 적용된 다중 AZ를 통해 고가용성을 지원합니다. Memcached 워크로드의 경우, 데이터가 클러스터의 모든 노드에 분할되므로 수요가 증가할 때 더 많은 데이터를 더 잘 처리하도록 확장할 수 있습니다. ElastiCache를 사용하면 하드웨어 프로비저닝, 소프트웨어 패치, 모니터링, 장애 복구 및 백업과 같은 관리 작업을 더 이상 수행할 필요가 없습니다. ElastiCache는 워크로드를 계속 실행하기 위해 클러스터를 지속적으로 모니터링하기 때문에 사용자는 애플리케이션 개발에 집중할 수 있습니다.

| 비교                            | Memcached | Redis |
|-------------------------------|-----------|-------|
| DB 부하를 오프로드하는 단순 캐시           | 예         | 예     |
| 쓰기/스토리지를 위해 수평적으로 확장할 수 있는 기능 | 예         | 아니요   |
| 다중 스레드 성능                     | 예         | 아니요   |
| 고급 데이터 유형                     | 아니요       | 예     |
| 데이터 세트 정렬/순위 지정               | 아니요       | 예     |
| Pub/Sub 기능                    | 아니요       | 예     |
| 자동 장애 조치가 있는 다중 가용 영역         | 아니요       | 예     |
| 지속성                           | 아니요       | 예     |

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



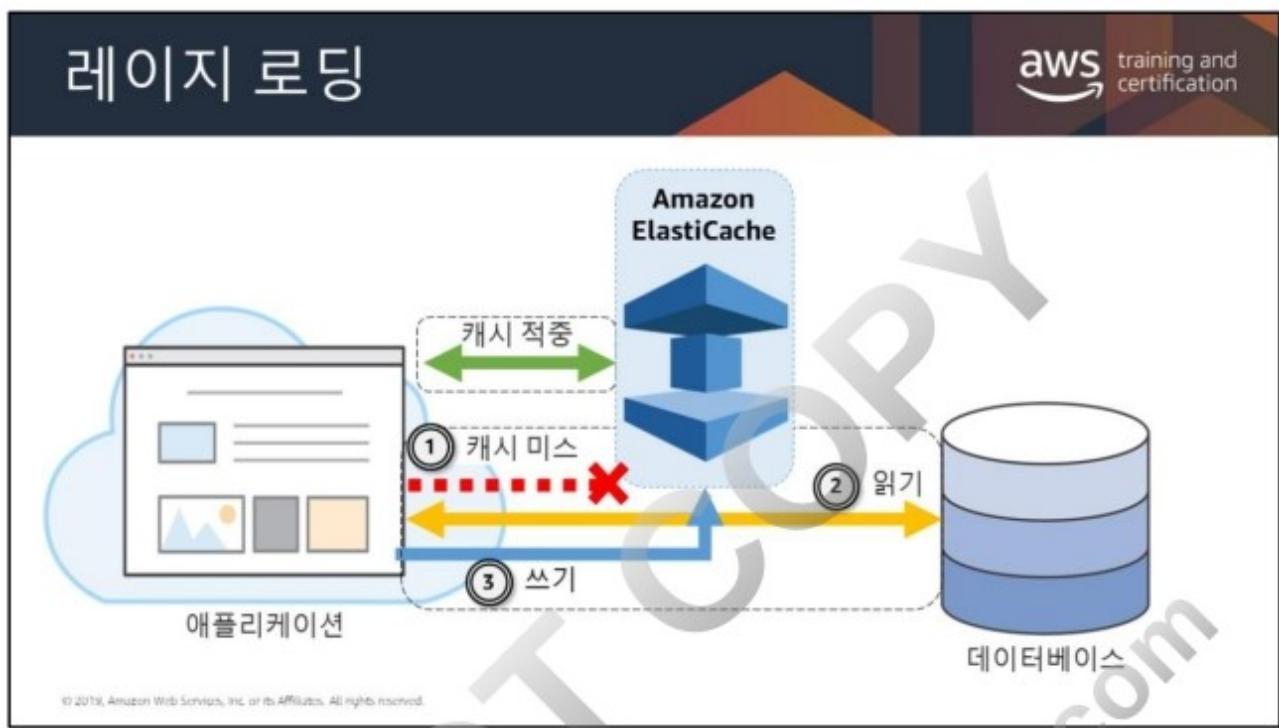
백엔드 데이터 스토어에 캐시를 사용하는 경우, 사이드 캐시가 가장 일반적으로 알려진 접근 방식일 것입니다. 정식 예에는 Redis와 Memcached가 모두 포함됩니다. 이러한 캐시는 기본 데이터 스토어와 분리된 범용 캐시이며, 워크로드 및 내구성 요구 사항에 따라 읽기 및 쓰기 처리량에 도움이 될 수 있습니다.

읽기 중심의 워크로드의 경우, 사이드 캐시는 일반적으로 다음과 같이 사용됩니다.

1. 지정된 키-값 쌍의 경우, 애플리케이션이 먼저 캐시에서 데이터 읽기를 시도합니다. 캐시가 데이터로 채워진 경우(캐시 적중), 해당 값이 반환됩니다. 그렇지 않은 경우, 2단계로 이동합니다.
2. 원하는 키-값 쌍을 캐시에서 찾을 수 없는 경우, 애플리케이션이 기본 데이터 스토어에서 데이터를 가져옵니다.
3. 애플리케이션이 데이터를 다시 가져와야 할 경우, 데이터가 존재하도록 2단계의 키-값 쌍이 캐시에 기록됩니다.

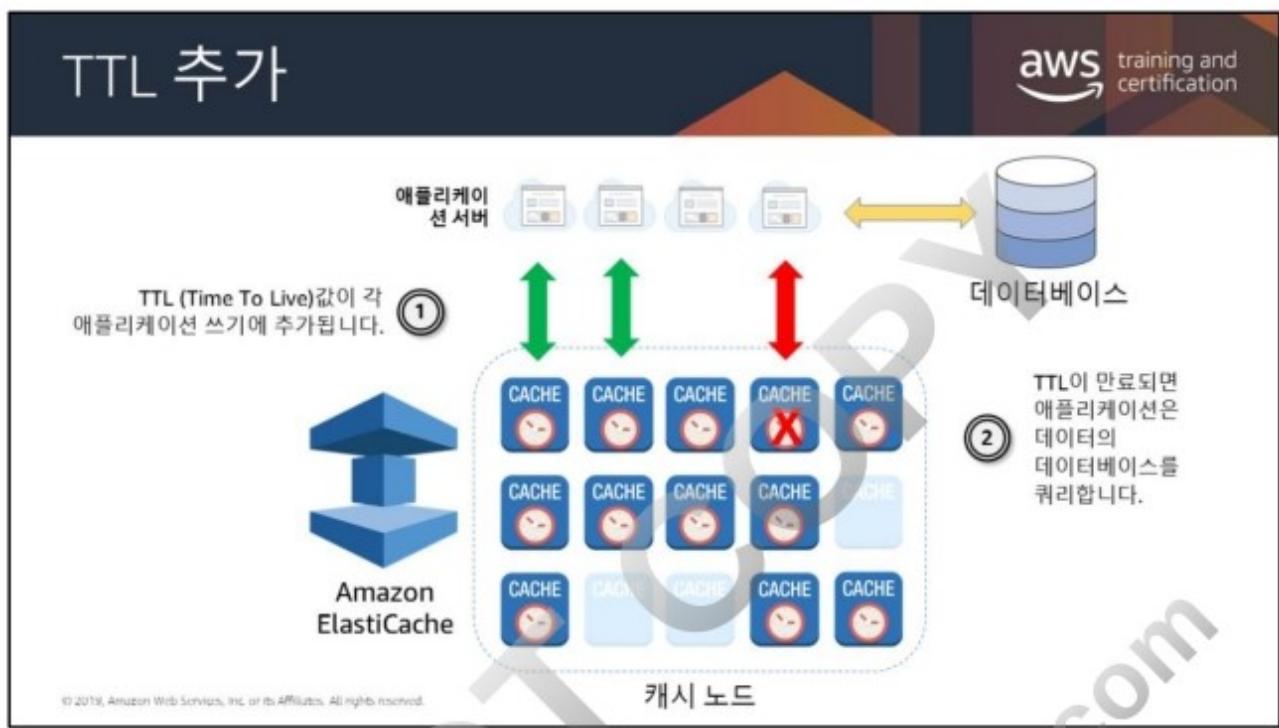
자세한 내용은 다음을 참조하십시오.

<https://aws.amazon.com/blogs/database/amazon-dynamodb-accelerator-dax-a-read-throughwrite-through-cache-for-dynamodb/>

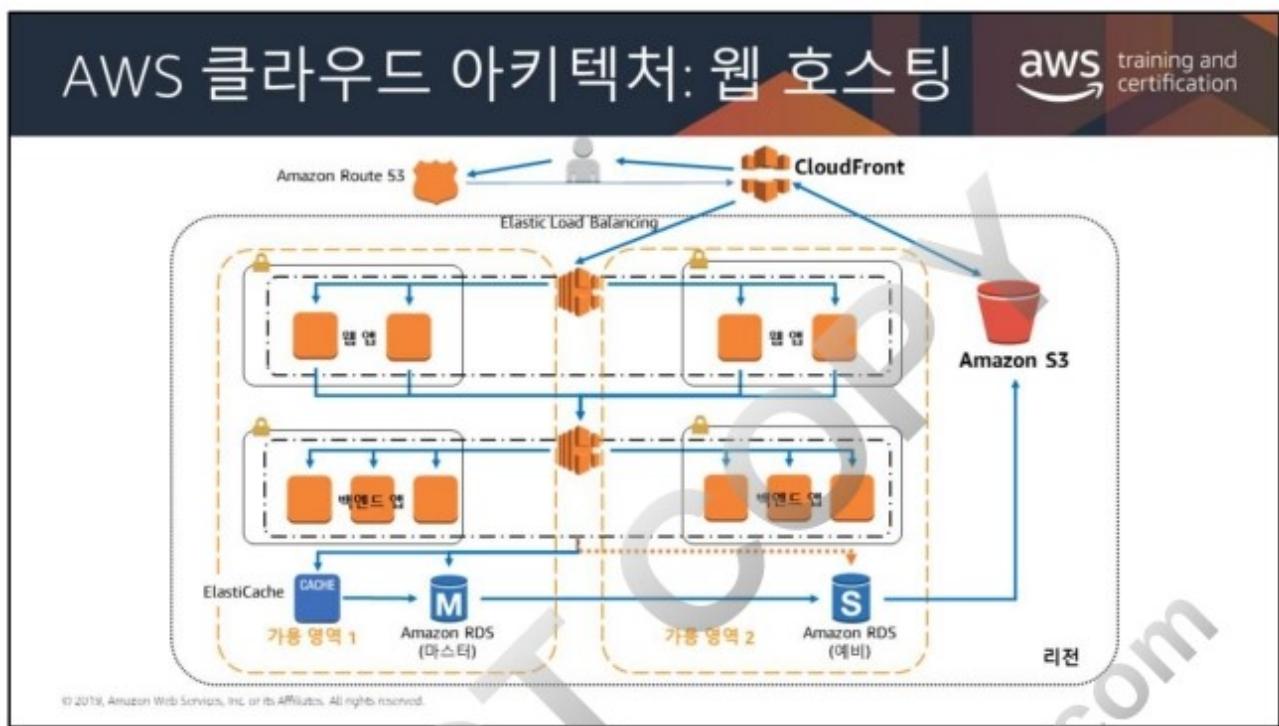


레이저 로딩은 필요할 때만 데이터를 캐시로 로드하는 캐싱 전략입니다. 이 배포에서 ElastiCache는 사용자의 애플리케이션과 액세스 대상의 데이터 스토어 또는 데이터베이스 사이에 위치합니다. 애플리케이션은 데이터를 요청할 때마다 먼저 ElastiCache 캐시로 요청을 보냅니다. 데이터가 캐시에 존재하며 최신일 경우, 캐시 적중이 발생하며 ElastiCache는 데이터를 애플리케이션으로 반환합니다. 그렇지 않으면 애플리케이션은 데이터를 애플리케이션에 반환하는 데이터 스토어에 데이터를 요청합니다. 이에 애플리케이션은 스토어에서 받은 데이터를 캐시에 작성합니다. 따라서 다음 번에 데이터 요청이 있을 때 해당 데이터를 좀 더 신속하게 검색할 수 있습니다.

레이저 로딩을 이용하면 요청된 데이터만 캐시됩니다. 한 번도 요청되지 데이터가 대부분이므로 레이저 로딩은 불필요한 데이터로 캐시를 채우는 상황을 방지할 수 있습니다. 다만 캐시 미스 페널티가 있습니다. 각각의 캐시 미스는 3회의 이동으로 나타납니다. 이로 인해 애플리케이션으로 데이터를 가져오는 작업이 눈에 띄게 지연될 수 있습니다. 또한 캐시 미스가 있을 때에만 데이터가 캐시에 작성될 경우, 데이터베이스에서 데이터가 변경될 때 캐시를 업데이트하지 않으므로 캐시의 데이터는 오래된 데이터가 될 수 있습니다. 이러한 문제를 해결하는 연속 쓰기 및 TTL 추가 전략에 대한 설명은 다음 시간에 다루기로 하겠습니다.



레이저 로딩은 기한 경과 데이터에 대해 허용되는 반면, 연속 쓰기는 데이터를 항상 최신 상태로 유지하며 다만 불필요한 데이터로 캐시를 채울 수 있습니다. TTL (Time To Live) 값을 각각의 쓰기에 추가하면 각 전략을 활용할 수 있으며 캐시를 데이터로 채우는 것을 대체로 방지할 수 있습니다. TTL은 키가 만료될 때까지 인 메모리 엔진에 따라 수 초 또는 수 밀리초의 수를 지정하는 정수값 또는 키입니다. 애플리케이션이 만료된 키를 읽으려고 시도할 때 이러한 시도는 캐시에서 데이터를 찾을 수 없는 것처럼 처리됩니다. 즉, 데이터베이스는 쿼리되며 캐시는 업데이트됩니다. 이렇게 하면 데이터를 기한 내에 업데이트할 수 있으며, 캐시의 값은 때때로 데이터베이스에서 새로 고쳐야 합니다.



기존 웹 호스팅 아키텍처는 아키텍처를 표시 계층, 애플리케이션 계층 및 지속성 계층으로 나눈 일반 3티어 웹 애플리케이션 모델을 구현합니다. 표시 계층, 애플리케이션 계층 또는 지속성 계층에서 호스트를 추가하면 확장성이 제공됩니다.

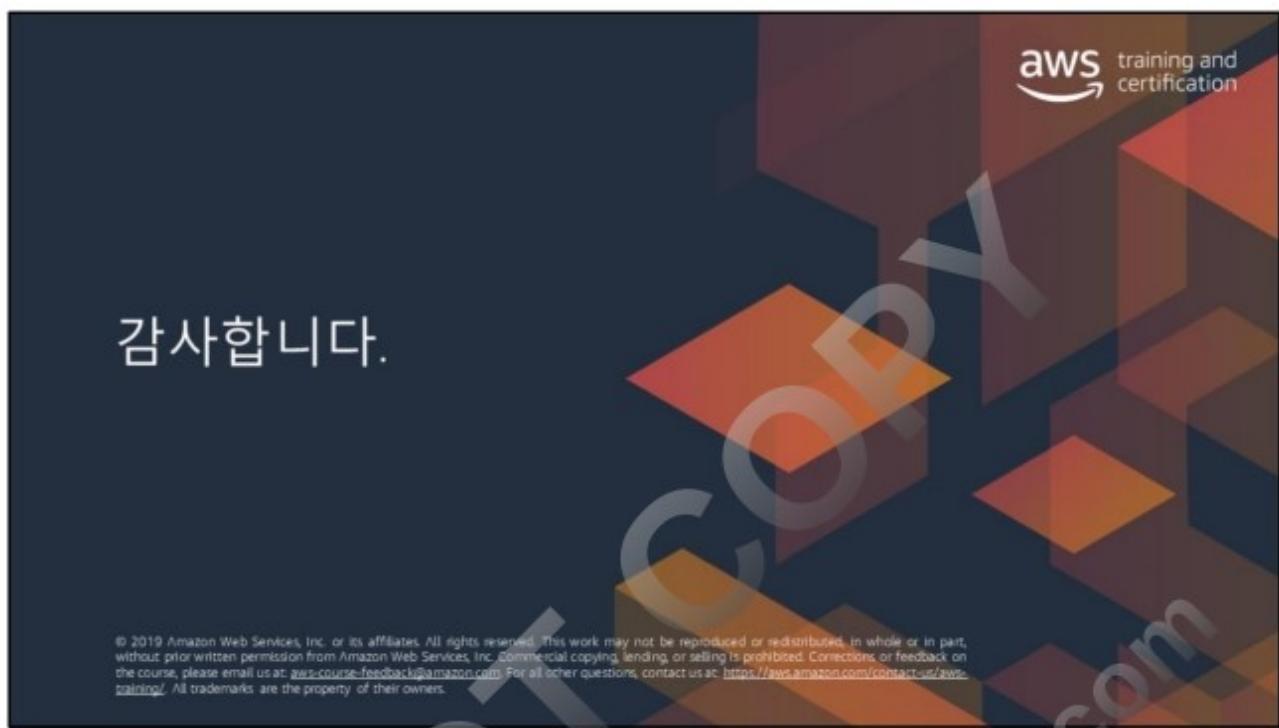
### AWS 클라우드에서 웹 애플리케이션 호스팅

기존 웹 호스팅 아키텍처는 약간의 수정만 거쳐도 AWS 제품에서 제공되는 클라우드 서비스로 손쉽게 이식할 수 있습니다. 하지만 여기서 제기해야 할 첫 번째 문제는 기존 웹 애플리케이션 호스팅 솔루션을 AWS 클라우드로 옮겼을 때 어떤 가치가 있으느냐 하는 것입니다. 클라우드가 적합한 솔루션이라고 판단될 경우, 적절한 아키텍처가 필요합니다.

- *Amazon Route 53*은 도메인 관리 및 Zone APEX 지원을 간소화하기 위한 DNS 서비스를 제공합니다.
- *Amazon CloudFront*는 대용량 콘텐츠를 위한 엣지 캐싱을 제공합니다.
- *Elastic Load Balancing*은 이 디아이그램의 웹 서버 Auto Scaling 그룹으로 트래픽을 분산시킵니다.
- 외부 방화벽은 보안 그룹을 통해 모든 웹 서버 인스턴스로 이동되었습니다.

- 백엔드 방화벽은 모든 백엔드 인스턴스로 이동되었습니다.
- Amazon EC2 인스턴스의 앱 서버 로드 밸런서는 앱 서버 클러스터 전반에 걸쳐 트래픽을 분산시킵니다.
- *Amazon ElastiCache*는 앱에 대한 캐싱 서비스를 제공하여 데이터베이스 티어에서 부하를 제거합니다.

DO NOT COPY  
zlagusdbs@gmail.com





## 모듈 11



### 아키텍처 측면에서의 필요성

이제 아키텍처는 수십만 명의 사용자들을 지원하지만 일부가 실패하면 전체 애플리케이션이 실패합니다. 종속성을 제거해야 합니다.

#### 모듈 개요

- 결합 해제된 아키텍처
- Amazon SQS 및 Amazon SNS를 사용하여 결합 해제된 아키텍처 구축

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



“밀결합”이라는 용어가 의미하는 것은 무엇입니까?

www.example.com

웹 티어

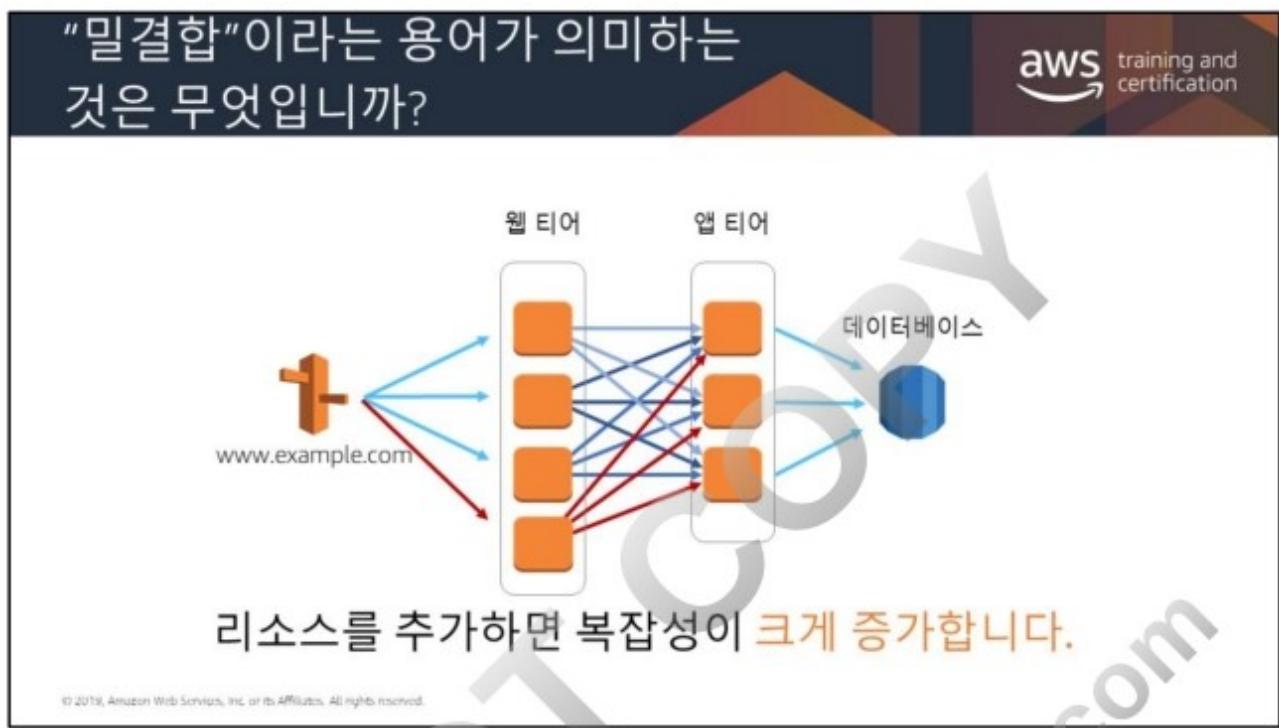
앱 티어

데이터베이스

구성 요소들은 서로 **강력하게 결합**되어 있습니다.

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

기존 인프라는 강력하게 통합된 서버 체인을 중심으로 움직이며 각 서버는 특정 목적을 가지고 있습니다. 이러한 구성 요소/계층의 하나에 장애가 발생하면 시스템에 치명적인 영향을 줄 수 있습니다. 또한, 이 때문에 규모 조정이 지연될 수 있습니다. 한 계층에 서버를 추가하거나 제거하는 경우, 연결된 모든 계층의 모든 서버가 적절하게 연결되어야 합니다.



구성 요소 하나의 변경이나 장애가 다른 구성 요소에 영향을 주지 않도록 상호 종속성을 줄여야 합니다. 느슨하게 결합된 경우, 관리형 솔루션을 시스템 계층 간의 중간자로 활용할 수 있습니다. 이렇게 하면 중간자가 구성 요소 또는 계층의 장애 및 규모 조정을 자동으로 처리합니다.

“밀결합”이라는 용어가 의미하는 것은 무엇입니까?

aws training and certification

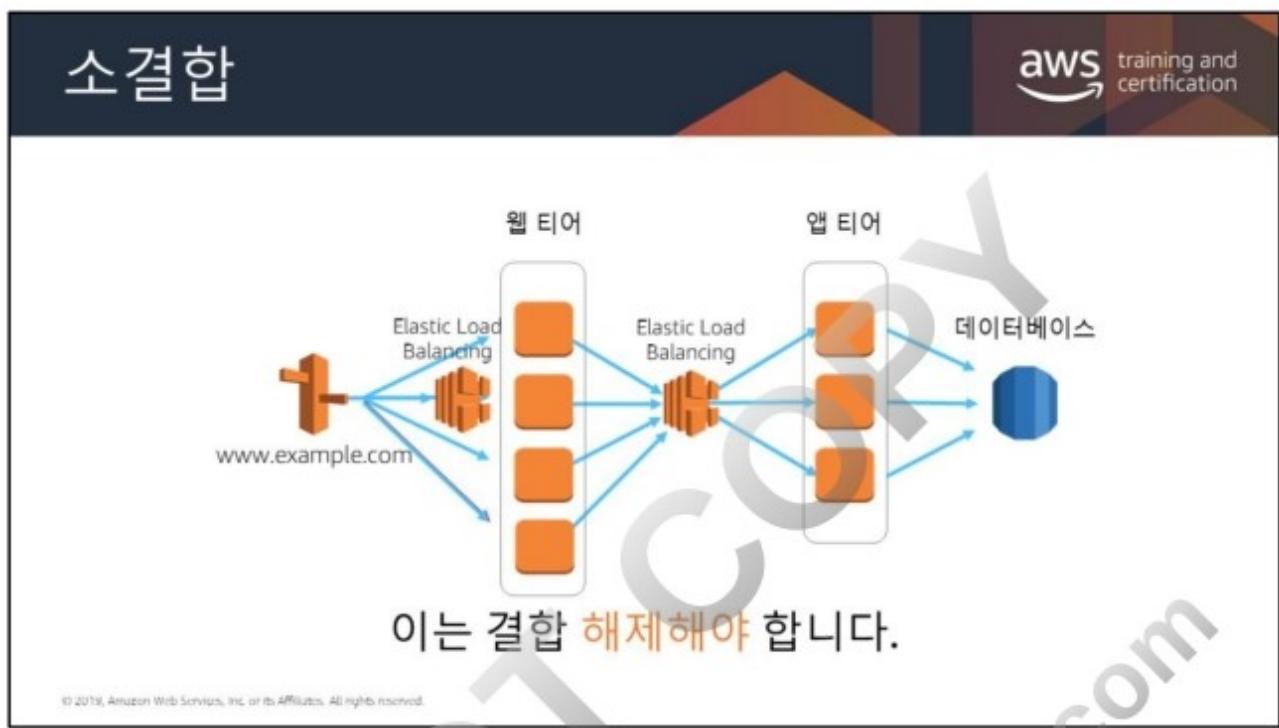
웹 티어      앱 티어

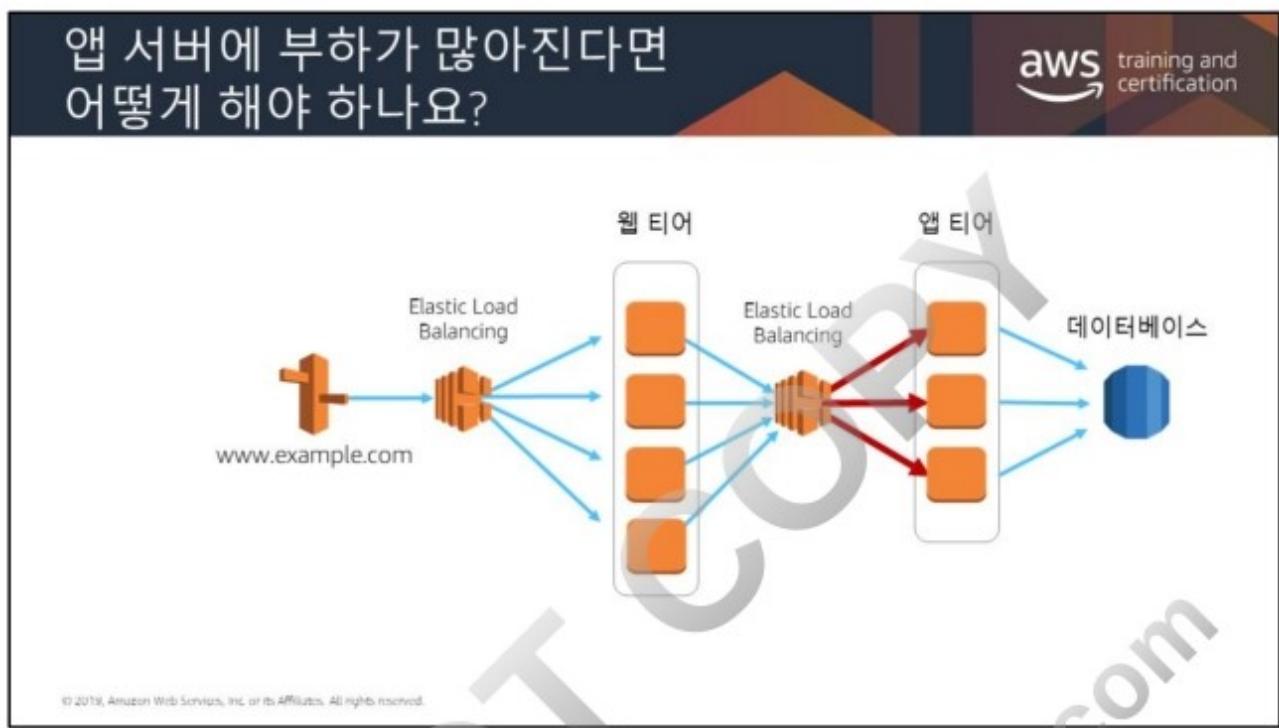
www.example.com

데이터베이스

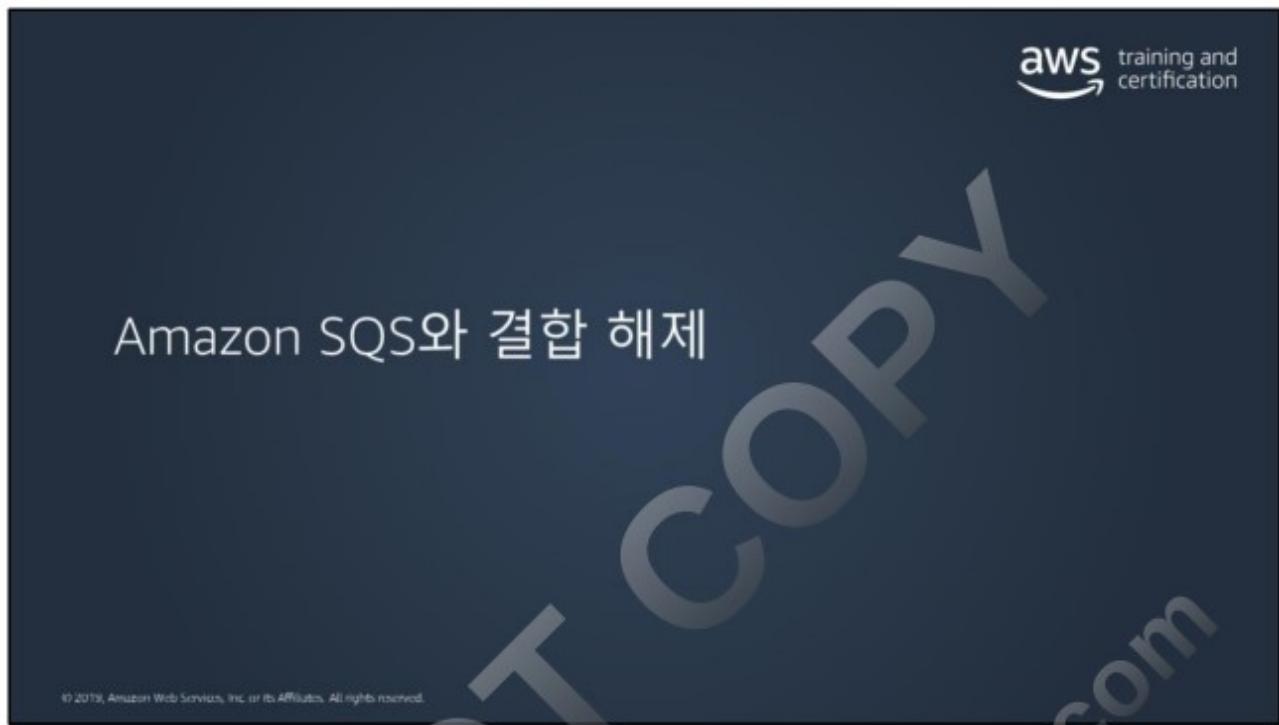
이는 결합 해제해야 합니다.

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.





고객 주문을 처리하는 웹 애플리케이션을 고려합니다. 주문 처리 워크플로우의 한 가지 잠재적인 취약점은 해당 주문을 데이터베이스에 저장하는 데 있습니다. 기업은 모든 주문이 데이터베이스에 계속 유지되는 것을 새로운 요구 사항으로 기대합니다. 그러나 잠재적 교착, 경합 조건 또는 네트워크 문제가 발생할 경우, 해당 주문을 계속 유지할 수 없게 됩니다. 결국 해당 주문은 복원을 위한 어떤 수단도 없이 손실됩니다.



## Amazon Simple Queue Service (Amazon SQS)



aws training and certification

완전 관리형 메시지 대기열 서비스

메시지는 처리 및 삭제될 때까지 저장됩니다.

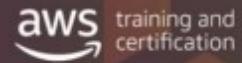
발신자와 수신자 간 버퍼 역할을 담당

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

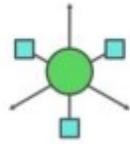
Amazon SQS는 관리 부담이 없으며 최소한의 구성으로도 바로 사용할 수 있는 완전 관리형 서비스입니다. 이 서비스는 방대한 규모로 작동하므로 하루에 수십억 건의 메시지를 처리할 수 있습니다. 컴퓨터, 네트워크 또는 가용 영역의 장애가 발생하더라도 메시지를 액세스할 수 있도록, 다수의 중복 가용 영역이 있는 고가용성의 단일 AWS 리전 내에 모든 메시지 대기열과 메시지를 저장합니다. 메시지는 동시에 전송하고 읽을 수 있습니다.

개발자는 Amazon SQS 대기열을 익명으로 또는 특정 AWS 계정과 안전하게 공유할 수 있습니다. IP 주소와 특정 시간으로 대기열 공유를 제한할 수도 있습니다. 서버 측 암호화(SSE)는 AWS Key Management Service (AWS KMS)에서 관리하는 키를 사용하여 Amazon SQS 대기열의 메시지 콘텐츠를 보호합니다. SSE는 Amazon SQS가 메시지를 수신하는 즉시 이를 암호화합니다. 이 메시지는 암호화된 형태로 저장되며 Amazon SQS는 권한이 있는 사용자에게 전송할 메시지만 복호화합니다.

## 소결합 실현(Amazon SQS 사용)



SQS 대기열을 사용하면 다음 사항들을 수행할 수 있습니다.

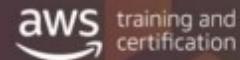


비동기식 처리를  
사용하여 각 단계에서  
신속하게 응답

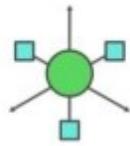
© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

DO NOT COPY  
zlagusdbs@gmail.com

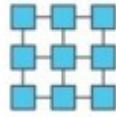
## 소결합 실현(Amazon SQS 사용)



SQS 대기열을 사용하면 다음 사항들을 수행할 수 있습니다.



비동기식 처리를  
사용하여 각 단계에서  
신속하게 응답

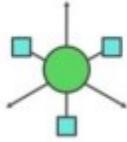
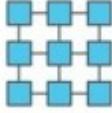
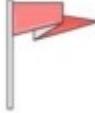


작업 인스턴스의 수를  
늘려 성능 및 서비스 요구  
사항 처리

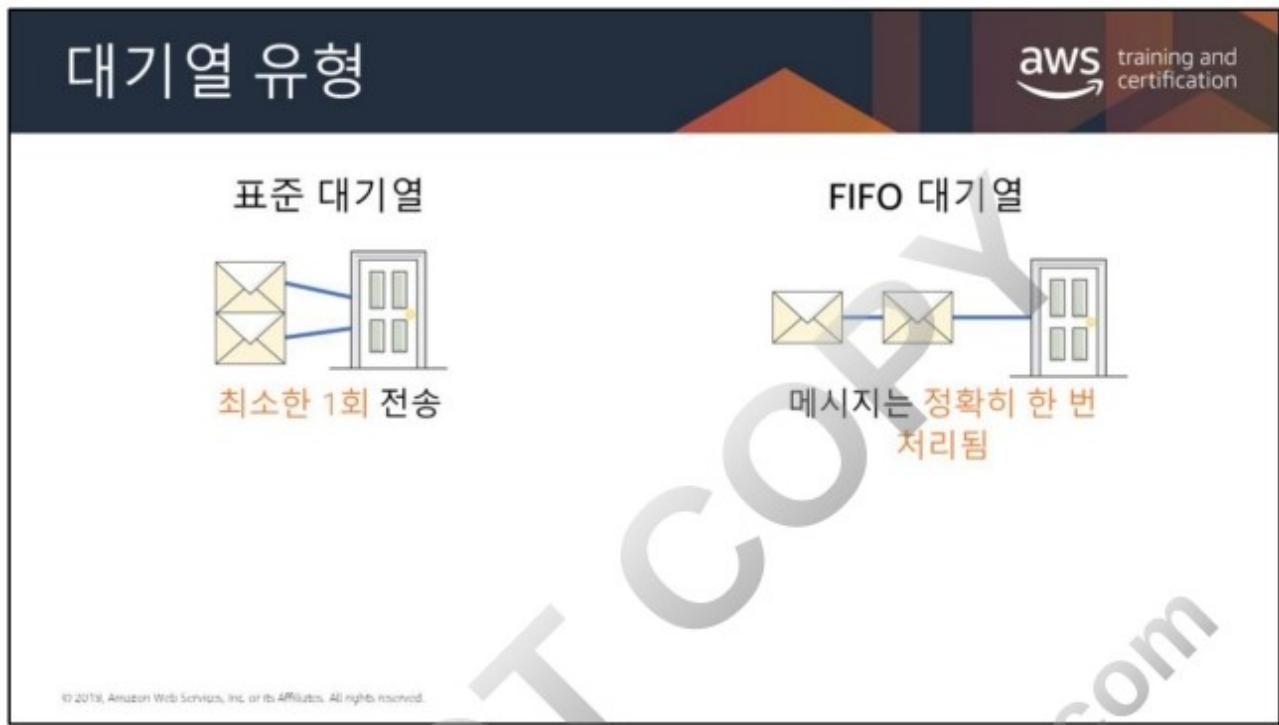
© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

## 소결합 실현(Amazon SQS 사용)

SQS 대기열을 사용하면 다음 사항들을 수행할 수 있습니다.

- 비동기식 처리를 사용하여 각 단계에서 신속하게 응답
- 작업 인스턴스의 수를 늘려 성능 및 서비스 요구 사항 처리
- 메시지가 대기열에 남아 있기 때문에 실패한 단계에서 쉽게 복구

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



SQS 대기열은 2가지 유형 즉, 표준 대기열과 FIFO 대기열로 구분됩니다.

**표준 대기열**은 최소 1회 전송 및 최선의 정렬을 제공합니다.

최소 1회 전송이란 때때로 메시지 사본이 2개 이상 전송되는 것을 의미합니다.

최선의 정렬은 때때로 메시지가 전송된 순서와는 다르게 전송될 수 있음을 의미합니다.

**FIFO(선입선출)** 대기열은 메시지가 전송된 순서대로 정확히 한 번 처리될 수 있도록 설계되어 있습니다.

## 대기열 유형

aws training and certification

표준 대기열

최소한 1회 전송

API 작업당 거의 무한한 수의  
초당 트랜잭션(TPS).

FIFO 대기열

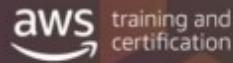
메시지는 정확히 한 번  
처리됨

초당 300건의 메시지까지 지원

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

다만 표준 대기열은 최대 처리량을 제공하는 반면, FIFO 대기열은 초당 최대 300건의 메시지(초당 300건의 전송, 수신 또는 삭제 작업)를 지원합니다. 작업당 최대 10건의 메시지를 일괄 처리할 경우, FIFO 대기열은 초당 최대 3,000건의 메시지를 지원할 수 있습니다.

## SQS 일반 사용 사례



The slide illustrates four common use cases for SQS:

- 작업 대기열**: Represented by a stack of three white squares on a green base.
- 버퍼 배치 작업**: Represented by a grid of colored squares (blue, magenta, yellow) with arrows indicating movement between them.
- 요청 오프로딩**: Represented by a circular icon with two overlapping arrows forming a cycle.
- 조정 트리거**: Represented by a bar chart with a pie chart on top, showing growth or distribution.

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

SQS 대기열을 사용하는 방법은 여러 가지가 있습니다.

**작업 대기열**: 동일한 양의 작업 일부를 동시에 처리하지 못할 수 있는 분산 애플리케이션의 구성 요소를 분리합니다.

**배치 작업 버퍼링**: 아키텍처에 확장성과 안정성을 더하고, 메시지를 손실하거나 지연 시간을 늘리지 않고 일시적인 볼륨 스파이크를 원활하게 처리합니다.

**요청 오프로딩**: 요청을 전송하여 대화식 요청 경로에서 속도가 느린 작업을 제거합니다.

**Auto Scaling**: Amazon SQS 대기열을 사용하면 애플리케이션의 로드를 결정하는데 도움이 됩니다. 또한 Auto Scaling과 결합 시 트래픽 볼륨에 따라 Amazon EC2 인스턴스의 수를 확장 또는 축소할 수 있습니다.