

DynamoDB

RDBMS

- RDBMS (Relational Database Management System) : 데이터의 관계에 집중하여 정형화된 형식으로 데이터를 관리하는 DB 관리 시스템
 - 행과 열로 구성된 2차원 테이블을 기반으로 정형화된 스키마 형식으로 데이터를 저장
- ACID 형태로 Transaction을 처리

ACID

- Transaction : 데이터베이스의 상태 변화를 수행하는 작업들을 묶은 단위
 - DB의 상태변화가 올바르게 이루어질 수 있도록 논리적으로 묶은 단위
 - 여러 작업이 한꺼번에 완전히 성공하거나, 전부 실패해야 함
 - 예) 내 계좌에서 돈 빠짐 + 친구계좌에 돈 들어감 → 이 둘이 한세트!
- Atomicity (원자성) : All or Nothing, 즉 전부 성공하거나 실패해야 함
- Consistency (일관성) : 트랜잭션 수행 전과 후에 일관적으로 데이터가 올바른 상태를 유지해야 함 → 실행 전후 데이터 상태가 논리적으로 맞아야 함
- Isolation (독립성) : 트랜잭션 간에는 서로 독립적으로 수행되어야 함
→ 동시에 여러 트랜잭션이 실행돼도 서로 간섭하지 않아야 함
- Durability (지속성) : 트랜잭션의 수행 결과는 영구적으로 데이터베이스에 반영되어야 함 → 디스크에 기록되어야 함!

RDBMS

- RDBMS (Relational Database Management System)
 - 데이터의 관계에 집중하여 정형화된 형식으로 데이터를 관리하는 DB 관리 시스템
 - 행과 열로 구성된 2차원 테이블을 기반으로 정형화된 스키마 형식으로 데이터를 저장
- ACID 형태로 Transaction을 처리
- 데이터 간 관계를 기반으로 데이터를 처리 가능
 - SQL, Join 등을 활용한 복잡한 쿼리 처리 가능
- 주요 AWS 서비스
 - Amazon RDS, Amazon Aurora, Amazon Redshift
- 주요 사용사례
 - 대부분의 어플리케이션의 메인 DB

NoSQL

- NoSQL (Not Only SQL) : 스키마 없이 다양한 형식의 데이터를 처리하는 데이터베이스 시스템
 - 별도로 정해진 스키마가 없으며 유연하게 데이터를 저장
- 종류
 - Key-Value Stores : Key-Value 쌍으로 데이터를 저장
 - Document Database : 데이터를 JSON 형식과 비슷한 일종의 문서 형식으로 데이터를 저장
 - Graph Database : 데이터 간의 연결을 중심으로 데이터를 저장

Key-Value Stores

Key

user:haeri:profile
user:haeri:recent_views
user:haeri:recommend
user:haeri:click_history
user:haeri:last_login

Value

age=39,gender=f,interests=tech,finance,travel
item_1042,item_902,item_381,item_777
item_123,item_784,item_999,item_1003
{"item_784":3,"item_123":1,"item_902":5}
2025-05-11T14:50:00+09:00

Document Database

```
{  
  "user_id": "u001",  
  "name": "Haeri",  
  "email": "haeri@example.com",  
  "age": 39,  
  "preferences": {  
    "genre": ["tech", "finance", "travel"],  
    "language": "ko"  
  },  
  "is_active": true  
}
```

Graph Database

- "사람(노드)"과 "관계(엣지)"를 중심으로 데이터를 저장
- 보통 SNS에서 사용
- 각 사용자는 하나의 노드로 / 노드 간의 관계(예: 친구, 팔로우, 좋아요 등)는 엣지(edge)로 저장

노드

```
{  
  "id": "jack",  
  "name": "Jack",  
  "type": "Person"  
}
```

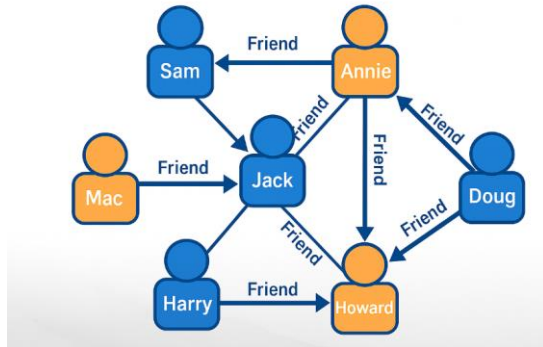
엣지

```
{  
  "from": "jack",  
  "to": "doug",  
  "type": "FRIEND"  
}
```

노드(Node): Sam, Annie, Jack, Doug, Mac, Harry, Howard
엣지

- Jack → Doug (친구)
- Jack → Annie (친구)
- Jack → Mac (친구)
- Jack → Harry (친구)
- Harry → Howard (친구)
- Sam → Annie (친구)
- Annie → Doug (친구)
- Doug → Howard (친구)
- Howard → Annie (친구)

Graph Databases:
데이터간의 연결을 중심으로 데이터를 저장



NoSQL

- NoSQL (Not Only SQL) : 스키마 없이 다양한 형식의 데이터를 처리하는 데이터베이스 시스템
 - 별도로 정해진 스키마가 없으며 유연하게 데이터를 저장
- 다양한 종류
- BASE 기반 처리

BASE

- 일반적으로 ACID에 비해서 훨씬 느슨하고 유연한 모델
- BAsically Available : 가용성을 중심으로 처리
(예: 가용성을 유지하기 위해 데이터를 다수 스토리지에 저장하며 접근 가능하게 유지)
- Soft-State : 데이터베이스의 상태는 느슨하고 유연함, 즉 확정된 상태로 계속 유지하지 않고 자연스럽게 원하는 상태로 수렴
- Eventual Consistency : 일관성이 바로 확보되지 않고 시간이 지나서언젠가 일관적인 상태로 수렴

NoSQL

- NoSQL (Not Only SQL) : 스키마 없이 다양한 형식의 데이터를 처리하는 데이터베이스 시스템
 - 별도로 정해진 스키마가 없으며 유연하게 데이터를 저장
- 다양한 종류
- BASE 기반 처리
- 가벼운 구조
 - 대부분 SQL 사용이 불가능하고 쿼리가 복잡하지 않고 간단한 처리만 가능
 - 확장성이 뛰어남
- 주요 AWS 서비스
 - Amazon DynamoDB, Amazon MemoryDB (Redis, memCached, ValKey) 등
- 주요 사용사례
 - 캐시, 간단한 데이터 저장, 추천 로직 등 각 DB 구조에 맞는 사용

정리!

항목	RDBMS	NoSQL
데이터 형식	정해진 스키마로 데이터 관리	자유로운 데이터 형식
데이터 처리	ACID	BASE
확장성	확장 어려움	비교적 분산 처리가 쉬움
쿼리 복잡도	복잡한 쿼리 수행 가능	비교적 간단한 쿼리만 수행 가능
AWS 서비스	RDS, Aurora, Redshift	MemoryDB, DocumentDB, DynamoDB 등

실습) DynamoDB를 생성해 보자!

 [DynamoDB](#) > 테이블

DynamoDB

<

대시보드

테이블

항목 탐색

PartiQL 편집기

백업

S3로 내보내기

S3에서 가져오기

통합 신규

예약 용량

설정

테이블 (0) [정보](#)

[모든 태그 키](#)

이름 ▲

상태

파티션 키

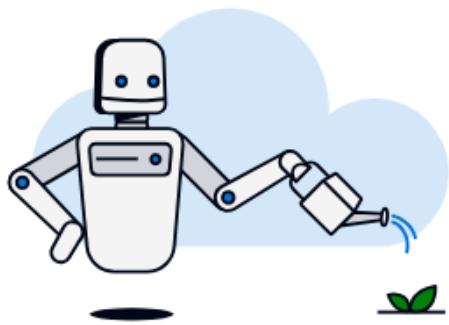
정렬 키

인덱스

복제 리전

삭제 방지

즐거찾



이 AWS 리전의 이 계정에는 테이블이 없

테이블 생성

DynamoDB Table 생성

- 테이블 이름 : sgu-202500-user-likes
- 파티션키 : user_id

☰ [DynamoDB](#) > [테이블](#) > 테이블 생성

테이블 생성

테이블 세부 정보 정보

DynamoDB는 테이블을 생성할 때 테이블 이름과 기본 키만 필요한 스키마를 생성합니다.

테이블 이름

테이블을 식별하는 데 사용됩니다.

sgu-202500-user-likes

문자, 숫자, 밑줄(_), 하이픈(-) 및 마침표(.)만 포함하는 3~255자의 문자입니다.

파티션 키

파티션 키는 테이블 기본 키의 일부로, 테이블에서 항목을 검색하고 확장성과 가용성을 위해 호스트에 데이터를 할당하는 데 사용되는 해시 값입니다.

user_id

1~255자이고 대소문자를 구분합니다.

정렬 키 - 선택 사항

정렬 키를 테이블 기본 키의 두 번째 부분으로 사용할 수 있습니다. 정렬 키를 사용하면 동일한 파티션 키를 공유하는 모든 항목을 정렬하거나 검색할 수 있습니다.

정렬 키 이름 입력

1~255자이고 대소문자를 구분합니다.

테이블 (1) 정보

🔍 테이블 찾기

모든 태그 키

모든

<input type="checkbox"/>	이름	상태	파티션 키	정렬 키	인덱스	복제 리전	삭제 방지	줄기
<input type="checkbox"/>	sgu-202500-user-likes	🔄 생성 중	user_id (S)	-	0	0	🔒 끄기	

Partition Key

- Partition Key : 모든 테이블에는 반드시 하나의 파티션 키가 있어야 함, 각 항목의 고유 ID 역할, **항목(Item)을 구분하고 저장 위치를 정하는 기준 키**
 - 같은 파티션 키 값끼리는 같은 물리 저장소 파티션에 묶임
 - 파티션 키는 내부적으로 해시(Hash) 처리를 거쳐, 데이터를 어떤 물리 파티션(서버)에 저장할지를 결정 함!
 - 예) minji01 → 해시 → 983248123912 → Partition A

개념	설명
파티션 키 해시	문자열을 숫자로 바꿔줌
저장 위치 결정	그 숫자에 따라 물리 파티션이 자동 선택됨
장점	분산 저장, 성능 균형 유지, 확장성 좋음

Partition Key

- DynamoDB는 user_id 같은 파티션 키를 숫자로 바꿔서 '어디에 저장할지'를 자동으로 정함!

그래서 어떤 데이터는 서버 1번에, 어떤 데이터는 서버 2번에, 이렇게 분산되어 저장됨 → 속도 빠름!

파티션 키 = 데이터를 어느 **서버 구역**에 넣을지 정하는 기준 키!

해시: 문자나 데이터를 고정된 길이의 "숫자"로 바꿔주는 계산 방식, 글자수가 많은 적든 정해진 길이! 즉 64자리(256비트) 길이의 고정된 값 생성 됨!

DynamoDB 데이터 삽입

DynamoDB

대시보드

테이블

항목 탐색

PartiQL 편집기

백업

S3로 내보내기

S3에서 가져오기

통합 [신규](#)

예약 용량

설정

테이블 (1)

모든 태그 키

모든 태그 값

테이블 찾기

sgu-202500-user-likes

sgu-202500-user-likes

▼ 항목 스캔 또는 쿼리

스캔 쿼리

테이블 또는 인덱스 선택

테이블 - sgu-202500-user-likes

속성 프로젝트 선택

모든 속성

필터 - 선택 사항

실행 재설정

완료됨 · 반환된 항목: 0 · 스캔한 항목: 0 · 효율성: 100% · 소비된 RCU: 2

테이블: sgu-202500-user-likes - 반환된 항목 (0)

스캔 시작 날짜: 5월 12, 2025, 04:29:09

항목 없음

표시할 항목이 없습니다.

항목 생성

JSON뷰 클릭

항목 생성

항목의 속성을 추가, 제거 또는 편집할 수 있습니다. 최대 32 수준까지 다른 속성 안에 속성을 중첩할 수 있습니다. [자세히 알아보기](#)

양식 | JSON 뷰

속성

새 속성 추가 ▼

항목 생성

항목의 속성을 추가, 제거 또는 편집할 수 있습니다. 최대 32 수준까지 다른 속성 안에 속성을 중첩할 수 있습니다. [자세히 알아보기](#)

속성 ☒ DynamoDB JSON 보기

json 입력 한 뒤,
항목생성 클릭

취소

항목 생성

```
1 {  
2   "user_id": { "S": "haeri05" },  
3   "product_name": { "S": "무선 이어폰" },  
4   "category": { "S": "디지털" },  
5   "reason": { "S": "최근 블루투스 기기 검색" },  
6   "timestamp": { "S": "2025-05-11T21:00:00" },  
7   "price": { "N": "89000" },  
8   "brand": { "S": "소니" },  
9   "color": { "S": "블랙" },  
10  "rating": { "N": "4.7" }  
11 }  
12
```

테이블: sgu-202500-user-likes - 반환된 항목 (1)

스캔 시작 날짜: 5월 12, 2025, 04:29:09



작업 ▼

항목 생성

< 1 > ⚙

<input type="checkbox"/>	user_id (문자열)	brand	category	color	price	product_name
<input type="checkbox"/>	haeri05	소니	디지털	블랙	89000	무선 이어폰

json file

이 json file을 차례로 입력하여
결과를 확인해 보자!

```
{
  "user_id": { "S": "haeri05" },
  "product_name": { "S": "무선 이어폰" },
  "category": { "S": "디지털" },
  "reason": { "S": "최근 블루투스 기기 검색" },
  "timestamp": { "S": "2025-05-11T21:00:00" },
  "price": { "N": "89000" },
  "brand": { "S": "소니" },
  "color": { "S": "블랙" },
  "rating": { "N": "4.7" }
}
```

```
{
  "user_id": { "S": "minji01" },
  "product_name": { "S": "핑크 가습기" },
  "category": { "S": "생활가전" },
  "timestamp": { "S": "2025-05-11T21:01:00" }
}
```

```
{
  "user_id": { "S": "jihoon07" },
  "product_name": { "S": "텀블러" }
}
```

```
{
  "user_id": { "S": "yuna23" },
  "product_name": { "S": "토끼 인형" },
  "reason": { "S": "장난감 카테고리 체류시간 높음" }
}
```

항목 생성 결과

테이블: sgu-202500-user-likes - 반환된 항목 (4)

스캔 시작 날짜: 5월 12, 2025, 04:29:09



작업 ▼

항목 생성

<input type="checkbox"/>	user_id (문자열) ▼	brand ▼	category ▼	color ▼	price ▼	product_name ▼	rating ▼	reason
<input type="checkbox"/>	yuna23					토끼 인형		장난감 카테..
<input type="checkbox"/>	jihoon07					텀블러		
<input type="checkbox"/>	minji01		생활가전			핑크 가습기		
<input type="checkbox"/>	haeri05	소니	디지털	블랙	89000	무선 이어폰	4.7	최근 블루투..

실습1) 항목 추가하기

- user_id : jihoon07
- product_name : 스탠리

추가 해 보자!

테이블: sgu-202500-user-likes - 반환된 항목 (4)									
스캔 시작 날짜: 5월 13, 2025, 11:28:22									
<input type="checkbox"/>	user_id (문자열)	brand	category	color	price	product_name	rating	reason	timestamp
<input type="checkbox"/>	jihoon07					텀블러			
<input type="checkbox"/>	yuna23					토끼 인형		장난감 카테...	
<input type="checkbox"/>	minji01		생활가전			빙크 가습기			2025-05-11T21:01:00
<input type="checkbox"/>	haeri05	소니	디지털	블랙	89000	무선 이어폰	4.7	최근 블루투...	2025-05-11T21:00:00

실습 2) 항목 편집

완료됨 · 반환된 항목: 4 · 스캔한 항목: 4 · 효율성: 100% · 소비된 RCU: 2

테이블: sgu-202500-user-likes - 반환된 항목 (1/4)
스캔 시작 날짜: 5월 13, 2025, 11:28:22

user_id (문자열) | brand | category | color | price | product_name | rating | reason | timestamp

<input checked="" type="checkbox"/>	jihoon07				텀블러				
<input type="checkbox"/>	yuna23				토끼 인형		장난감 카테...		
<input type="checkbox"/>	minji01	생활가전			핑크 가습기			2025-05-11T21:01:00	

항목 편집
항목 복제
항목 삭제
선택한 항목을 CSV로 다운로드
결과를 CSV로 다운로드

작업 ▲

항목 생성

항목 편집

양식 | JSON 뷰

항목의 속성을 추가, 제거 또는 편집할 수 있습니다. 최대 32 수준까지 다른 속성 안에 속성을 중첩할 수 있습니다. [자세히 알아보기](#)

속성

새 속성 추가 ▼

속성 이름	값	유형	
user_id - 파티션 키	jihoon07	문자열	
product_name	스탠리	문자열	제거

취소

저장

저장 후 닫기

지금 방식의 문제는?

- 파티션 키가 있을 때는 덮어쓴다!
- 파티션 키만 사용하여 한 명당 하나의 항목만 저장!
- 그래서 여러 개 저장하려면!!!!
- 정렬 키 (Sort Key) 필요!
- 그럼 생성된 테이블에 정렬키 추가 가능할까?

실습 3) 정렬키 추가한 테이블 생성!

- 테이블 명 : sgu-202500-user-likes-time
- 파티션 키 : user_id, 문자열
- 정렬 키 : timestamp, 문자열
- 근데.. timestamp는 날짜타입 아닌가?
 - DynamoDB는 날짜 타입이 없음!
 - ISO 8601 형식이면 시간순 정렬도 가능!
 - 실제 의미는 "날짜/시간"이지만, 타입은 문자열로 취급

실습 4) 생성된 테이블에 항목추가하기!

```
{
  "user_id": {
    "S": "haeri05"
  },
  "timestamp": {
    "S": "2025-05-11T21:00:00"
  },
  "product_name": {
    "S": "무선 이어폰"
  },
  "category": {
    "S": "디지털"
  },
  "reason": {
    "S": "최근 블루투스 기기 검색"
  },
  "price": {
    "N": "89000"
  },
  "brand": {
    "S": "소니"
  },
  "color": {
    "S": "블랙"
  },
  "rating": {
    "N": "4.7"
  }
}
```

```
{
  "user_id": {
    "S": "minji01"
  },
  "timestamp": {
    "S": "2025-05-11T21:01:00"
  },
  "product_name": {
    "S": "핑크 가습기"
  },
  "category": {
    "S": "생활가전"
  }
}
```

```
{
  "user_id": {
    "S": "jihoon07"
  },
  "timestamp": {
    "S": "2025-05-12T10:15:00"
  },
  "product_name": {
    "S": "스탠리 텀블러"
  }
}
```

```
{
  "user_id": {
    "S": "yuna23"
  },
  "timestamp": {
    "S": "2025-05-12T11:20:00"
  },
  "product_name": {
    "S": "토끼 인형"
  },
  "reason": {
    "S": "장난감 카테고리 체류시간 높음"
  }
}
```

테이블: sgu-202500-user-likes-time - 반환된 항목 (4)

스캔 시작 날짜: 5월 13, 2025, 11:46:50

<input type="checkbox"/>	user_id (문자열)	timestamp (문자열)	brand	category	color	price	product_name	rating	reason
<input type="checkbox"/>	jihoon07	2025-05-12T10:15:00					스탠리 텀블러		
<input type="checkbox"/>	yuna23	2025-05-12T11:20:00					토끼 인형		장난감 카테고리 체류시간 높음
<input type="checkbox"/>	minji01	2025-05-11T21:01:00		생활가전			핑크 가습기		
<input type="checkbox"/>	haeri05	2025-05-11T21:00:00	소니	디지털	블랙	89000	무선 이어폰	4.7	최근 블루투스 기기 검색

실습 5) 항목 추가하기

- user_id : jihoon07
- timestamp : 2025-05-12T18:15:00
- product_name : 텀블럭 악세사리

실습 6) 항목 정렬하기!

- jihoon07를 시간 순으로 정렬해보자!

sgu-202500-user-likes-time

▼ 항목 스캔 또는 쿼리

☐ 스캔

☒ 쿼리

테이블 또는 인덱스 선택

테이블 - sgu-202500-user-likes-time

속성 프로젝트 선택

모든 속성

파티션 키: user_id

jihoon07

정렬 키: timestamp

가림

정렬 키 값 입력

☐ 내림차순 정렬

▼ 필터 - 선택 사항

속성 이름

Q 속성 이름 입력

조건

가림

유형

문자열

값

속성 값 입력

제거

필터 추가

실행

재설정