굿모닝아이텍 신입 기초 질문 150 + 자기소개·지원동기 + 데이터 용어

1 차 실무진 면접 대비 · 2026-11-24

이 질문집의 기준

- **잡플래닛 면접 후기 90 건에서 실제로 나온 질문**을 중심에 두고, 공고 필수역량(Linux, 네트워크, SQL, API)과 1 지망 Denodo·2 지망 Cloudera 의 입문 수준 질문을 더했습니다. 후기에 실제로 나온 질문은 **(후기)**로 표시했습니다.
- 답은 **30 초 분량(2~3 문장)**입니다. 첫 문장이 정의, 둘째 문장이 핵심 특징이나 예시입니다. 더 물으면 그때 깊이 들어가면 됩니다.
- 기초 질문은 **틀리지 않고 깔끔하게** 말하는 게 중요합니다. 후기에도 "기본 이상의 꼼꼼한 답을 원했다"는 말이 있습니다.
- 자기소개서·프로젝트 꼬리질문은 [면접답변집](면접답변집_굿모닝아이텍.md)에, 깊은 원리는 [필수정리](Denodo_Cloudera_필수정리.md)에 있습니다.

0. 면접 시작 대본: 자기소개와 지원동기

면접은 거의 항상 이 두 질문으로 시작합니다. **외워서 읽지 말고, 굵은 글씨의 핵심 문장만 기억한 뒤 자기 말로** 하세요. 숫자와 경험은 모두 제출한 서류에 적힌 사실입니다.

0-1. 1 분 자기소개

> 안녕하십니까, **흩어진 데이터를 잇는 일을 해 온 지원자 김태동입니다.**
> 공공 SI 에서 약 11 개월 동안 지방세 가상계좌 시스템과 지자체 단속 시스템을 개발하고 운영했습니다. 가상계좌 조회가 느려졌을 때는 짐작하지 않고 실행계획부터 확인해, 파티셔닝과 인덱스를 다시 설계해 **조회 시간을 400~500ms 에서 50~60ms 로 줄였습니다.** 단속 사진 전송 장애는 민원을 기다리지 않고 실패 건부터 모아 원인을 찾아, **월 15 건이던 장애 민원을 0 건으로** 만들었습니다.
> 단속 시스템에서는 Delphi 메인 시스템과 Python 모듈, 전자결재, 차량조회 API 를 잇는 일을 맡으면서, **데이터가 시스템마다 흩어져 있을 때 생기는 문제**를 직접 겪었습니다.
> 지금은 SK AX SKALA 과정에서 Docker 와 Kubernetes 로 서비스를 배포·운영하며 **고객 인프라 위에 플랫폼을 올리는 기본기**를 다지고 있습니다. 배운 것과 겪은 실패는 문서로 남겨 팀이 같은 실수를 반복하지 않게 하는 습관이 있습니다.
> 이 경험들을 바탕으로 **굿모닝아이텍에서 데이터 솔루션 엔지니어로 성장하고 싶습니다.**

(약 55 초. 첫 문장, 숫자 두 개, 마지막 문장만 정확하면 나머지는 자기 말로 해도 됩니다.)

짧게 해 달라고 할 때 (30 초)

> 공공 SI 에서 약 11 개월 동안 백엔드 개발과 운영을 하며, 실행계획으로 조회 시간을 85% 줄이고 서로 다른 시스템과 API 를 잇는 일을 했습니다. 지금은 SKALA 에서 Docker 와 Kubernetes 로 서비스를 배포·운영하고 있습니다. 흩어진 데이터를 잇던 경험을 살려, 굿모닝아이텍에서 데이터 솔루션 엔지니어로 성장하고 싶습니다.

0-2. 지원동기 (데이터 솔루션 엔지니어로 성장하고 싶은 이유)

> **결론부터 말씀드리면, 데이터 솔루션 엔지니어로 제대로 성장할 수 있는 곳이 굿모닝아이텍이라고 생각해 지원했습니다.**

> 공공 SI 에서 가장 손이 많이 간 일은 시스템마다 흩어진 데이터를 연동 지점마다 코드로 잇고 오류를 보정하는 일이었습니다. 그때 데이터를 옮기지 않고 여러 원천을 하나로 묶는 데이터 가상화를 알게 되면서, **이 문제를 개별 코드가 아니라 데이터 플랫폼으로 푸는 엔지니어가 되고 싶다**는 목표가 생겼습니다. 저는 고객의 데이터가 어디에 있든, 그 데이터를 믿고 쓸 수 있게 만드는 사람이 되고 싶습니다.
>
> 그런 엔지니어로 성장하려면 세 가지가 필요하다고 봤습니다. 첫째, **좋은 데이터 솔루션을 깊이 다룰 기회**입니다. 굿모닝아이텍은 Cloudera 국내 유일의 골드 파트너이고, Denodo 와 Snowflake 까지 함께 다룹니다. 둘째, **데이터 플랫폼이 올라가는 인프라에 대한 이해**입니다. 가상화와 클라우드, 보안까지 고객 인프라 전체를 다루는 회사라서, 데이터만 아는 엔지니어가 아니라 인프라까지 아는 엔지니어로 클 수 있습니다. 셋째, **사람을 키우는 문화**입니다. 신입 교육과 자격증 취득에 꾸준히 투자하는 회사라는 점이 오래 다니며 성장하고 싶은 이유입니다.
>
> 입사하면 먼저 Denodo 와 Cloudera 를 깊이 익혀 설치와 연계를 혼자 맡을 수 있게 되고, 1 년 안에 공식 자격을 따겠습니다. 처음 맡는 구축은 설치부터 장애 처리까지 문서로 남겨 팀의 자산으로 만들겠습니다. 길게는 **고객의 데이터 구조를 듣고 어떤 플랫폼으로 풀지 설계할 수 있는 데이터 솔루션 엔지니어**로 성장해, 굿모닝아이텍이 키우는 데이터 사업에 보탬이 되겠습니다.

(약 1 분 10 초. 시간이 짧으면 "세 가지" 문단을 한 줄씩만 말합니다.)

말할 때 주의할 점

- "성장하고 싶다"만 반복하면 회사를 학교로 본다는 인상을 줄 수 있습니다. **성장해서 무엇을 보탤지**(설치·연계를 혼자 맡기, 문서로 팀 자산 만들기, 데이터 사업 기여)를 꼭 붙입니다.
- 회사 정보(Cloudera 골드 파트너, Denodo·Snowflake, 교육 투자)는 **지원 이유의 근거**로만 짧게 씁니다. 회사 자랑을 길게 늘어놓지 않습니다.
- 인재상과 연결: 끝까지 책임지는 태도(장애 민원 0 건)는 **책임**, 새 솔루션을 원리부터 공부하는 모습은 **열정**, 문서로 팀을 돕는 습관은 **배려**입니다. 단어로 말하지 않아도 이야기 속에 드러납니다.

0-3. 이어서 나올 꼬리질문

질문	답의 뼈대
개발자가 아니라 엔지니어로 가려는 이유는?	SI 에서 가장 보람 있던 일이 새 기능 개발보다 남이 만든 시스템을 잇고, 운영 중 문제를 끝까지 해결하는 일이었습니다(장애 민원 0 건, 무중단 계좌 풀 반영). 데이터 솔루션 엔지니어는 그 일을 고객 인프라 위에서 플랫폼으로 하는 직무라 가장 잘 맞는다고 생각합니다. 개발

	경험은 API 연동과 자동화 스크립트에 그대로 쓰입니다.
데이터 솔루션 엔지니어로 성장한 모습은 구체적으로 어떤가요?	3 년 뒤에는 고객의 DB·DW·API 구조를 듣고 Denodo 로 묶을지 Cloudera 에 적재할지 판단해 구축을 이끌 수 있는 엔지니어, 그다음에는 후배에게 설치부터 장애 대응까지 가르칠 수 있는 엔지니어가 되는 것입니다.
다른 데이터 회사도 많은데 왜 우리인가요?	데이터 제품 하나만 파는 회사가 아니라, Cloudera·Denodo·Snowflake 를 함께 다루면서 그 아래 가상화·클라우드·보안까지 다루는 회사는 드뭅니다. 인프라까지 아는 데이터 엔지니어로 크기에 가장 맞는 곳입니다.
성장하고 나면 더 큰 회사로 옮기지 않을까요?	첫 회사는 방향을 찾는 과정에서 나왔고, 지금은 데이터 솔루션 엔지니어라는 방향이 분명합니다. Denodo 와 Cloudera 는 고객 구축 경험이 쌓여야 깊어지는 일이라, 한 회사에서 오래 경험을 쌓는 것이 제 성장에도 가장 유리하다고 생각합니다.
회사가 신입에게 투자하는 걸 어떻게 알았나요?	회사 소개와 언론 보도에서 신입 교육과 직무 교육에 꾸준히 투자한다는 내용을 봤습니다. 정부 지원 교육 수료생을 채용해 온 회사라는 점도 SKALA 과정을 밟고 있는 저에게 의미 있게 다가왔습니다.

1. 회사와 솔루션 (가장 많이 나옴)

#	질문	30 초 답
1	굿모닝아이텍은 어떤 회사인가요? (후기)	VMware, Dell, Cloudera, Denodo, Akamai 같은 글로벌 IT 인프라 솔루션을 기업과 공공기관에 공급하고, 구축부터 운영까지 기술지원을 하는 회사입니다. VMware 가상화로 성장해 지금은 클라우드, 보안, 빅데이터·AI 로 넓혔고, 2025 년 매출은 1,300 억 원대, 직원의 80% 이상이 엔지니어입니다.
2	우리 솔루션을 아나요? VMware 를 아나요? (후기)	VMware 는 서버 한 대를 여러 가상 서버로 나눠 쓰게 해 주는 가상화 솔루션입니다. ESXi 로 서버를 나누고, vCenter 로 여러 서버를 묶어 vMotion, HA, DRS 로 무중단 운영을 합니다. 굿모닝아이텍은 국내 VMware 최고 매출 파트너상을 여러 번 받았습니다.
3	VMware 를 다뤄 봤나요? (후기)	직접 운영해 본 적은 없습니다. 대신 SKALA 에서 Kubernetes 로 서비스를 배포하고 운영하면서 가상화·컨테이너의 기본기를 익혔고, VMware 구조와 기능을 공부했습니다. 입사하면 빠르게 익히겠습니다.
4	우리 회사의 데이터 사업을 아나요?	Cloudera 국내 유일 골드 파트너이고, 2023 년부터 Denodo 파트너로 데이터 가상화 사업을, 최근에는 Snowflake 까지 하고 있습니다. AI 전략도 Cloudera AI 플랫폼 공급 확대로 정했다고 알고 있습니다.
5	우리 회사의 자체 제품을 아나요?	Clovir 시리즈가 있습니다. 사설 클라우드 포털(ClovirSM, ClovirONE), 쿠버네티스 컨테이너 관리 포털(ClovirCM), VDI 통합 관리(ClovirVDI)입니다. ClovirCM 은 2024 년 GS 인증을 받았습니다.
6	데이터 솔루션 엔지니어는 무슨 일을 한다고 생각하나요?	고객사 환경에 Denodo 나 Cloudera 를 설치하고, 고객의 DB·DW·API 를 연계하고, 성능과 보안을 설정한 뒤, 운영 중 장애를 지원하는 일입니다. 제품을 만드는 일이 아니라 고객 인프라 위에 올리고 책임지는 일이라고 이해하고 있습니다.
7	엔지니어 직무가 하는 일은? (후기)	솔루션을 고객 환경에 맞게 설계·구축하고, 운영 중 문제가 생기면 원인을 찾아 해결하며, 고객에게 기술을 설명하는 일입니다. 기술력과 함께 고객 응대와 문서화가 중요하다고 생각합니다.

2. 가상화·클라우드·VDI (후기에서 가장 많은 기술 주제)

#	질문	30 초 답
8	가상화란? (후기)	하나의 물리 자원을 소프트웨어로 나눠 여러 개의 독립된 자원처럼 쓰게 하는 기술입니다. 서버 한 대를 여러 VM 으로 나누면 남는 자원을 활용하고, 서버 수와 비용을 줄이고, 장애 때 VM 을 옮기거나 다시 띄울 수 있습니다.
9	가상화의 종류는? (후기)	서버 가상화(VMware vSphere), 데스크톱 가상화(VDI, Horizon), 네트워크 가상화(NSX), 스토리지 가상화(vSAN)가 대표적입니다. 애플리케이션을 격리하는 컨테이너도 넓게는 가상화 기술로 봅니다.
10	하이퍼바이저란? Type 1 과 Type 2 의 차이는?	물리 하드웨어를 여러 VM 에 나눠 주는 소프트웨어입니다. Type 1 은 서버에 직접 설치하는 방식(VMware ESXi, Hyper-V)으로 기업 서버용이고, Type 2 는 Windows 같은 OS 위에 프로그램으로 설치하는 방식(VirtualBox)으로 개인 PC 용입니다.
11	클라우드와 가상화의 차이는? (후기)	가상화는 자원을 나누는 기술이고, 클라우드는 그 자원을 셀프서비스·종량제·탄력적 확장 같은 서비스 형태로 제공하는 방식입니다. 가상화는 클라우드를 만드는 바탕 기술 중 하나입니다.
12	프라이빗·퍼블릭·하이브리드 클라우드의 차이는? (후기)	프라이빗은 한 기업 전용으로 자기 데이터센터에 구축한 클라우드, 퍼블릭은 AWS·Azure 처럼 여러 고객이 함께 빌려 쓰는 클라우드, 하이브리드는 둘을 함께 쓰는 구성입니다. 보안과 규제가 중요한 금융·공공은 프라이빗이나 하이브리드를 많이 씁니다.
13	IaaS, PaaS, SaaS 의 차이는? (후기)	제공사가 어디까지 관리하느냐의 차이입니다. IaaS 는 서버·네트워크 같은 인프라를(AWS EC2), PaaS 는 OS 와 실행 환경까지를(AWS RDS), SaaS 는 완성된 소프트웨어까지를(Microsoft 365) 제공합니다. 위로 갈수록 운영 부담은 줄고 통제권도 줄어듭니다.
14	VDI 란? (후기)	사용자 PC 환경을 데이터센터 서버의 VM 으로 돌리고, 사용자는 화면만 전송받아 쓰는 데스크톱 가상화입니다. 데이터가 PC 에 남지 않아 보안에 좋고, 중앙에서 관리할 수 있습니다. VMware Horizon(현재 Omnissa)이 대표 제품입니다.
15	망분리와 VDI 는 무슨 관계인가요?	금융·공공은 규제상 인터넷망과 업무망을 나눠야 하는데, PC 를 두 대 두는 대신 VDI 로 한쪽 망을 가상 PC 로 쓰는 논리적 망분리가 많이 쓰였습니다. 굿모닝아이텍이 크게 성장한 배경입니다.
16	프로비저닝 방식을 설명해 보세요. (후기)	디스크를 만들 때 공간을 언제 잡느냐의 차이입니다. Thick 은 처음에 전체 용량을 미리 잡아 성능이 안정적이고, Thin 은 실제로 쓴 만큼만 잡아 공간을 아끼지만 여러 VM 이 동시에 늘면 데이터스토어가 꽉 찰 위험이 있습니다.

17	VM 과 컨테이너의 차이는?	VM 은 하이퍼바이저 위에서 OS 를 통째로 따로 가지고, 컨테이너는 호스트 OS 커널을 공유하고 애플리케이션과 라이브러리만 격리합니다. 그래서 컨테이너가 가볍고 빨리 뜨지만, 격리 수준은 VM 이 더 강합니다.
18	vMotion, HA, DRS 를 설명해 보세요.	vMotion 은 실행 중인 VM 을 끄지 않고 다른 서버로 옮기는 기능, HA 는 서버가 죽으면 그 위의 VM 을 다른 서버에서 자동으로 재시작하는 기능, DRS 는 서버 간 부하를 보고 VM 을 자동으로 분산하는 기능입니다.
19	스냅샷은 백업인가요?	아닙니다. 스냅샷은 특정 시점으로 되돌리기 위한 변경 기록이라, 원본 디스크가 망가지면 같이 잃습니다. 오래 두면 파일이 커지고 성능이 떨어집니다. 백업은 별도 저장소에 복사본을 두는 것입니다.

3. 서버·스토리지·백업 기초 (후기)

#	질문	30 초 답
20	서버를 설명해 보세요. (후기)	네트워크로 다른 컴퓨터에 서비스(웹, DB, 파일 등)를 제공하는 컴퓨터입니다. 일반 PC 보다 CPU·메모리가 크고, 전원·디스크를 이중화해 24 시간 안정적으로 돌도록 만듭니다. 랙에 꽂는 랙 서버가 데이터센터에서 가장 흔합니다.
21	RAID 란? 0, 1, 5, 6, 10 의 차이는?	여러 디스크를 묶어 성능이나 안정성을 높이는 기술입니다. 0 은 나눠 쓰기(빠르지만 하나만 고장 나도 전부 잃음), 1 은 거울 복사, 5 는 패리티로 디스크 1 개 고장까지 견딤(최소 3 개), 6 은 2 개 고장까지 견딤(최소 4 개), 10 은 미러링과 나눠 쓰기를 함께 씁니다.
22	DAS, NAS, SAN 의 차이는?	DAS 는 서버에 직접 붙인 디스크, NAS 는 네트워크로 파일 단위로 공유하는 저장소(NFS, SMB), SAN 은 전용 네트워크로 블록 단위로 연결하는 저장소(FC, iSCSI)입니다. VMware 공유 스토리지로는 SAN 이나 NAS, 또는 vSAN 을 씁니다.
23	백업이란? (후기)	데이터를 원본과 다른 저장소에 복사해 두었다가, 장애·실수·랜섬웨어로 원본을 잃었을 때 복구하는 것입니다. 사본 3 개, 매체 2 종류, 1 개는 외부 보관이라는 3-2-1 원칙이 기본입니다.
24	Veeam 이 무엇인가요? (후기)	VMware 같은 가상화 환경의 백업·복구에 강한 소프트웨어입니다. VM 을 통째로 백업하고 빠르게 복구할 수 있습니다. 굿모닝아이텍이 공급하는 제품 중 하나입니다.
25	RPO 와 RTO 는?	RPO 는 장애 때 잃어도 되는 데이터의 시간 범위(예: 최근 1 시간), RTO 는 복구까지 걸려도 되는 시간입니다. 둘 다 짧을수록 비용이 많이 듭니다.
26	HCI 란?	서버, 스토리지, 네트워크를 별도 장비 없이 x86 서버 여러 대에 소프트웨어로 통합한 구성입니다. VMware vSAN 이 대표적이고, 서버만 추가하면 용량과 성능이 함께 늘어납니다.

4. Linux 기초 (후기: 기본 명령어, 부족하면 테스트)

4-1. 반드시 바로 말할 명령어

명령	뜻	명령	뜻
`pwd`	현재 위치	`ls -al`	숨김 파일까지 자세히 목록
`cd /경로`, `cd ..`, `cd ~`	이동, 상위로, 홈으로	`mkdir -p a/b`	하위까지 디렉터리 만들기
`cp -r a b`	복사(디렉터리는 -r)	`mv a b`	이동·이름 바꾸기
`rm -rf 경로`	강제·하위까지 삭제(주의)	`touch 파일`	빈 파일 만들기
`cat`, `less`	파일 보기	`head`, `tail -n 100`	앞부분, 뒷부분 보기
`tail -f 로그`	로그 실시간 보기	`grep -i "error" 파일`	문자열 찾기
`find / -name "*.log"`	파일 찾기	`chmod 755 파일`	권한 바꾸기
`chown user:group 파일`	소유자 바꾸기	`ps -ef`, `ps aux`	프로세스 목록
`top`	실시간 자원 사용량	`kill PID`, `kill -9 PID`	정상 종료, 강제 종료
`df -h`	디스크 용량	`du -sh 경로`	디렉터리 크기
`free -h`	메모리	`systemctl status·restart 서비스`	서비스 상태·재시작
`journalctl -u 서비스`	서비스 로그	`ss -tlnp` (예전 `netstat`)	열린 포트
`ping`, `curl`	연결 확인, HTTP 요청	`ssh user@host`, `scp`	원격 접속, 원격 복사
`tar -czvf a.tar.gz 경로` / `tar -xzvf`	압축 / 풀기	`sudo`	관리자 권한으로 실행
`whoami`, `id`	나는 누구	`history`	명령 기록
`man 명령`	설명서	`vi 파일`	편집기(아래

			표)

vi 최소한: `i` 입력 모드 → `Esc` 명령 모드 → `:wq` 저장 후 종료, `:q!` 저장 없이 종료, `/단어` 검색, `dd` 한 줄 삭제

4-2. 질문

#	질문	30 초 답
27	절대 경로와 상대 경로의 차이는?	절대 경로는 `/`부터 시작하는 전체 경로(`/var/log/messages`), 상대 경로는 현재 위치 기준 경로(`../logs`)입니다. 스크립트나 cron 에서는 실행 위치가 달라질 수 있어 절대 경로를 씁니다.
28	권한 755 는 무슨 뜻인가요?	소유자 rwx(7), 그룹 r-x(5), 그 외 r-x(5)입니다. r=4, w=2, x=1 을 더한 값입니다. 실행 파일이나 디렉터리에 흔히 쓰고, 설정 파일은 보통 644 나 640 을 씁니다.
29	로그를 실시간으로 보려면?	`tail -f 로그파일`을 쓰고, 에러만 보려면 `tail -f 로그 | grep -i error` 처럼 grep 과 함께 씁니다. systemd 서비스면 `journalctl -u 서비스 -f` 도 됩니다.
30	특정 프로세스를 찾아서 종료하려면?	`ps -ef | grep 이름`으로 PID 를 찾고 `kill PID` 로 정상 종료합니다. 안 끝나면 마지막 수단으로 `kill -9` 를 씁니다. -9 는 정리 작업 없이 끝내서 데이터가 꼬일 수 있기 때문입니다.
31	디스크가 꽉 찼을 때 어떻게 확인하나요?	`df -h` 로 어느 파티션이 찼는지 보고, `du -sh` 로 큰 디렉터리를 따라 내려가 원인(대개 로그)을 찾습니다. 운영 중인 로그는 바로 지우지 않고 압축하거나 비웁니다.
32	서버가 느리다면 무엇부터 보나요?	`top` 으로 CPU·메모리를 많이 쓰는 프로세스와 load average 를 보고, `free -h` 로 메모리, `df -h` 로 디스크를 확인합니다. 그다음 해당 프로세스의 로그를 봅니다.
33	서비스를 재시작하고 부팅 때 자동으로 뜨게 하려면?	`systemctl restart 서비스`로 재시작하고, `systemctl enable 서비스`로 자동 시작을 켭니다. 상태와 에러는 `systemctl status` 와 `journalctl -u` 로 봅니다.
34	포트가 열려 있는지 확인하려면?	서버 안에서는 `ss -tlnp` 로 어떤 프로세스가 어떤 포트를 듣는지 보고, 다른 서버에서는 `nc -zv 호스트 포트`나 `telnet` 으로 접속되는지 확인합니다.
35	root 로 직접 작업하면 안 되는 이유는?	실수 한 번이 시스템 전체에 영향을 주기 때문입니다. 평소에는 일반 계정을 쓰고, 필요할 때만 `sudo` 로 권한을 빌리면 기록도 남습니다.

5. 네트워크 기초 (후기: OSI, NAT·PAT)

#	질문	30 초 답
36	OSI 7 계층을 설명해 보세요. (후기)	통신 과정을 7 단계로 나눈 모델입니다. 1 물리(케이블), 2 데이터링크(MAC, 스위치), 3 네트워크(IP, 라우터), 4 전송(TCP·UDP, 포트), 5 세션, 6 표현(암호화·인코딩), 7 응용(HTTP, DNS)입니다. 실제 인터넷은 이를 4 계층으로 줄인 TCP/IP 모델을 씁니다.
37	계층별 대표 장비는?	L1 허브·케이블, L2 스위치(MAC 으로 전달), L3 라우터(IP 로 경로 결정), L4 로드밸런서(IP·포트로 분산), L7 L7 스위치·웹방화벽(HTTP 내용까지 봄)입니다.
38	TCP 와 UDP 의 차이는?	TCP 는 연결을 맺고 순서와 전달을 보장해 신뢰성이 높고(웹, DB 연결), UDP 는 연결 없이 빠르게 보내지만 보장하지 않습니다(DNS 조회, 스트리밍).
39	3-way handshake 란?	TCP 연결을 맺는 과정입니다. 클라이언트가 SYN 을 보내고, 서버가 SYN+ACK 로 답하고, 클라이언트가 ACK 를 보내면 연결이 성립합니다.
40	IP 주소, 서브넷 마스크, 게이트웨이는?	IP 주소는 장비의 주소, 서브넷 마스크는 그중 어디까지가 네트워크 부분인지 표시하는 값, 게이트웨이는 다른 네트워크로 나갈 때 거치는 라우터 주소입니다. 같은 서브넷끼리는 직접, 다르면 게이트웨이를 거칩니다.
41	사설 IP 와 공인 IP 의 차이는?	공인 IP 는 인터넷에서 유일한 주소, 사설 IP 는 내부망에서만 쓰는 주소(10.x, 172.16~31.x, 192.168.x)입니다. 사설 IP 는 NAT 를 거쳐 인터넷으로 나갑니다.
42	NAT 와 PAT 의 차이는? (후기)	NAT 는 사설 IP 를 공인 IP 로 바꿔 주는 기술이고, PAT 는 여러 사설 IP 가 공인 IP 하나를 함께 쓰면서 포트 번호로 구분하는 방식입니다. 가정용 공유기가 PAT 입니다.
43	DNS 란?	도메인 이름을 IP 주소로 바꿔 주는 시스템입니다. 먼저 /etc/hosts 를 보고, 없으면 DNS 서버에 물어 답을 일정 시간(TTL) 캐시합니다.
44	DHCP 란?	네트워크에 들어온 장비에 IP 주소, 서브넷 마스크, 게이트웨이, DNS 를 자동으로 나눠 주는 프로토콜입니다. 서버는 보통 바뀌지 않게 고정 IP 를 씁니다.
45	HTTP 와 HTTPS 의 차이는?	HTTPS 는 HTTP 를 TLS 로 암호화한 것입니다. 서버 인증서로 상대가 진짜인지 확인하고, 주고받는 데이터를 암호화합니다. 기본 포트는 HTTP 80, HTTPS 443 입니다.
46	자주 쓰는 포트 번호를 말해 보세요.	SSH 22, DNS 53, HTTP 80, HTTPS 443, MySQL 3306, Oracle 1521, PostgreSQL 5432, MSSQL 1433, RDP 3389 입니다.
47	방화벽이란?	정해진 규칙(출발지·목적지 IP, 포트, 프로토콜)에 따라 통신을 허용하거나

		막는 장비나 소프트웨어입니다. 서버 자체 방화벽과 네트워크 구간 방화벽이 따로 있어서, 둘 다 열어야 통신됩니다.
48	ping 은 되는데 서비스 접속이 안 되면?	네트워크 경로는 살아 있으니, 포트를 확인합니다. 서비스가 그 포트에서 떠 있는지(`ss -tlnp`), 방화벽이 그 포트를 막는지 봅니다. 반대로 ping 이 막혀 있어도 서비스는 될 수 있어서 ping 만으로 판단하지 않습니다.
49	로드밸런서란?	여러 서버 앞에서 요청을 나눠 보내 부하를 분산하고, 죽은 서버는 빼는 장비나 소프트웨어입니다. IP·포트로 나누는 L4 와 URL·헤더까지 보는 L7 이 있습니다.
50	VLAN 이란?	스위치 하나를 논리적으로 여러 네트워크로 나누는 기술입니다. 물리 장비를 바꾸지 않고 부서별·용도별로 망을 분리할 수 있습니다. VMware 에서도 포트 그룹에 VLAN 을 지정합니다.

6. DB·SQL 기초 (공고 필수역량)

#	질문	30 초 답
51	DB 와 DBMS 의 차이는?	DB 는 정리해서 저장한 데이터 자체이고, DBMS 는 그 데이터를 저장·조회·관리하는 소프트웨어(Oracle, MySQL, PostgreSQL)입니다.
52	SQL 명령의 종류는?	DDL(CREATE, ALTER, DROP: 구조 정의), DML(SELECT, INSERT, UPDATE, DELETE: 데이터 조작), DCL(GRANT, REVOKE: 권한), TCL(COMMIT, ROLLBACK: 트랜잭션)입니다.
53	기본 키와 외래 키는?	기본 키(PK)는 행을 유일하게 식별하는 컬럼으로 중복과 NULL 을 허용하지 않고, 외래 키(FK)는 다른 테이블의 기본 키를 참조해 테이블 간 관계를 지키는 컬럼입니다.
54	INNER JOIN 과 OUTER JOIN 의 차이는? (후기, 개발 직무)	INNER JOIN 은 양쪽 모두에 짝이 있는 행만, LEFT OUTER JOIN 은 왼쪽 테이블의 행을 모두 남기고 짝이 없으면 NULL 로 채웁니다. 예를 들어 주문이 없는 고객까지 보려면 LEFT JOIN 을 씁니다.
55	인덱스란? 장단점은?	책의 색인처럼 원하는 행을 빨리 찾게 해 주는 자료 구조(대개 B+트리)입니다. 조회는 빨라지지만, 저장 공간을 쓰고 INSERT·UPDATE 때 인덱스도 고쳐야 해서 쓰기는 느려집니다.
56	트랜잭션과 ACID 는?	트랜잭션은 모두 성공하거나 모두 취소돼야 하는 작업 묶음입니다. 원자성(전부 아니면 전무), 일관성(규칙 유지), 격리성(서로 간섭 안 함), 지속성(커밋되면 남음)을 보장합니다. 계좌 이체가 대표 예입니다.
57	정규화란?	중복을 줄이고 이상 현상(삽입·수정·삭제 오류)을 막기 위해 테이블을 나누는 과정입니다. 너무 나누면 조인이 많아져 느려질 수 있어서, 조회 성능을 위해 일부러 합치는 반정규화도 합니다.
58	뷰(View)란?	실제 데이터 없이 SELECT 문을 저장해 둔 가상 테이블입니다. 복잡한 쿼리를 단순하게 쓰고, 보여 줄 컬럼만 노출해 보안에도 씁니다. Denodo 가 여러 DB 에 걸친 '뷰'를 만드는 것과 같은 개념의 확장입니다.
59	GROUP BY 와 HAVING 은? WHERE 와 차이는?	GROUP BY 는 같은 값끼리 묶어 집계하고, HAVING 은 묶은 결과에 조건을 겁니다. WHERE 는 묶기 전 행에, HAVING 은 묶은 후 그룹에 거는 조건입니다.
60	RDBMS 와 NoSQL 의 차이는?	RDBMS 는 정해진 스키마의 표와 SQL, 트랜잭션이 강점이고,

		NoSQL(MongoDB, Redis, HBase)은 스키마가 유연하고 수평 확장이 쉬워 대용량·비정형 데이터에 맞습니다.
61	실행 계획이란?	DB 가 SQL 을 어떤 순서와 방법(인덱스를 탈지, 어떤 조인을 쓸지)으로 실행할지 세운 계획입니다. 느린 쿼리는 실행 계획부터 보고 고칩니다. 가상계좌 조회를 이렇게 개선했습니다.

7. 데이터 기초 (Denodo·Cloudera 입문)

#	질문	30 초 답
62	데이터 가상화란?	여러 곳에 흩어진 데이터를 옮기지 않고, 그 위에 하나의 논리 계층을 만들어 데이터베이스 하나처럼 조회하게 하는 기술입니다. 빠르게 제공하고, 최신 데이터를 보며, 보안을 한 곳에서 관리할 수 있습니다. Denodo 가 대표 제품입니다.
63	ETL 이란?	원천에서 데이터를 뽑아(Extract) 가공하고(Transform) DW 같은 저장소에 적재하는(Load) 과정입니다. 순서를 바꿔 적재 후 가공하는 방식을 ELT 라고 합니다.
64	데이터 가상화와 ETL 의 차이는?	ETL 은 데이터를 복사해 옮기고, 가상화는 옮기지 않고 조회할 때 가져옵니다. 가상화는 빠르고 최신이지만 원천에 부하가 가고, ETL 은 무거운 가공과 이력 분석에 강합니다. 실제로는 함께 씁니다.
65	DW(데이터 웨어하우스)와 데이터 레이크의 차이는?	DW 는 정제된 정형 데이터를 분석용으로 모은 저장소로 정합성이 좋고, 데이터 레이크는 정형·비정형 원본 데이터를 그대로 모아 두는 저장소로 유연하지만 품질 관리가 어렵습니다.
66	OLTP 와 OLAP 의 차이는?	OLTP 는 주문·이체처럼 짧은 거래를 많이 처리하는 업무 시스템, OLAP 은 대량 데이터를 집계·분석하는 분석 시스템입니다.
67	빅데이터의 특징(3V)은?	양(Volume), 속도(Velocity), 다양성(Variety)입니다. 여기에 정확성(Veracity), 가치(Value)를 더해 5V 라고도 합니다.
68	Hadoop 이란?	평범한 서버 여러 대를 묶어 대용량 데이터를 나눠 저장(HDFS)하고 나눠 처리하는 오픈소스 플랫폼입니다. Cloudera 는 이를 기업용으로 묶어 공급합니다.
69	HDFS 란?	Hadoop 의 분산 파일 시스템입니다. 파일을 128MB 블록으로 나눠 여러 서버에 3 벌씩 복제해 저장하고, NameNode 가 위치를 관리합니다. 서버 하나가 고장 나도 데이터가 남습니다.
70	Spark 란?	대용량 데이터를 여러 서버의 메모리에서 나눠 처리하는 분산 처리 엔진입니다. 디스크를 많이 쓰던 MapReduce 보다 빠르고, 배치·SQL·스트리밍·머신러닝을 모두 합니다.
71	Kafka 란?	대량의 이벤트·로그를 실시간으로 받아 보관하고 여러 시스템에 전달하는 분산 메시징 플랫폼입니다. SKALA 프로젝트에서 채팅 메시지 저장에 써 봤습니다.
72	Hive 와 Impala 의 차이는?	둘 다 Hadoop 데이터를 SQL 로 조회하지만, Hive 는 대량 배치 처리용이고 Impala 는 BI 도구의 빠른 대화형 조회용입니다.
73	BI 도구란?	Tableau, Power BI 처럼 데이터를 대시보드와 리포트로 시각화해 현업의 의사결정을 돕는 도구입니다. 보통 JDBC·ODBC 로 DW 나

		Denodo 에 접속합니다.
74	데이터 거버넌스란?	데이터의 품질, 보안, 권한, 출처(리니지)를 회사 차원의 규칙으로 관리하는 것입니다. 누가 어떤 데이터를 볼 수 있고, 그 데이터를 믿을 수 있는지를 보장합니다.
75	메타데이터란?	데이터에 대한 데이터입니다. 테이블 이름, 컬럼 타입, 만든 사람, 출처 같은 정보입니다.

8. 보안·인증 기초

#	질문	30 초 답
76	인증과 인가의 차이는?	인증은 "누구인지" 확인하는 것(로그인), 인가는 "무엇을 할 수 있는지" 권한을 정하는 것입니다. 인증 다음에 인가를 합니다.
77	AD(Active Directory)·LDAP 이란? (후기)	회사의 사용자·그룹·컴퓨터 정보를 한 곳에서 관리하는 디렉터리 서비스입니다. LDAP 은 이 정보를 조회하고 인증하는 표준 프로토콜이고, AD 는 Microsoft 의 구현입니다. 여러 시스템이 AD 계정 하나로 로그인하게 할 수 있습니다.
78	SSO 란?	한 번 로그인하면 여러 시스템을 다시 로그인하지 않고 쓰는 방식입니다. 사내망은 Kerberos, 웹·SaaS 는 SAML 이나 OIDC 로 구현합니다.
79	대칭키와 비대칭키 암호화의 차이는?	대칭키는 암호화와 복호화에 같은 키를 써서 빠르고, 비대칭키는 공개키와 개인키를 나눠 써서 키를 안전하게 나누기 좋습니다. HTTPS 는 비대칭키로 대칭키를 나눠 갖고, 실제 데이터는 대칭키로 암호화합니다.
80	제로트러스트란?	"아무도 믿지 말고 항상 확인하라"는 보안 원칙입니다. 내부망이라도 매번 인증·권한을 확인하고, 침입당했다고 가정해 내부 확산을 막습니다(마이크로 세그멘테이션). 굿모닝아이텍이 Akamai Guardicore 로 키우는 사업입니다.
81	랜섬웨어 대비 방법은?	정기 백업과 백업본 격리(3-2-1, 변경 불가 백업), 패치, 최소 권한, AD 계정 보호, 내부 확산 차단입니다.

9. 컨테이너·DevOps·모니터링 기초

#	질문	30 초 답
82	Docker 란?	애플리케이션과 실행 환경을 이미지로 묶어, 어디서나 같은 방식으로 컨테이너로 실행하게 하는 도구입니다. "내 PC 에선 되는데 서버에선 안 돼요" 문제를 줄입니다.
83	Kubernetes 란?	여러 서버에 걸쳐 컨테이너를 배포·확장·복구하는 오케스트레이션 플랫폼입니다. 원하는 상태를 선언하면 맞춰 주고, 죽은 컨테이너를 다시 띄웁니다. SKALA 에서 k3d·Helm·Argo CD 로 서비스 4 개를 운영해 봤습니다.
84	CI/CD 란?	CI 는 코드를 합칠 때마다 자동으로 빌드·테스트하는 것, CD 는 통과한 결과를 자동으로 배포하는 것입니다. AutoTC 에서 Jenkins 와 Argo CD 로 구성했습니다.
85	Git 이란?	코드의 변경 이력을 관리하는 분산 버전 관리 도구입니다. 브랜치로 나눠 작업하고, PR 로 검토한 뒤 합칩니다.
86	ELK 가 무엇의 약자인가요? (후기)	Elasticsearch(로그 저장·검색), Logstash(수집·가공), Kibana(시각화)입니다. 로그를 모아 검색하고 대시보드로 보는 스택입니다. 저는 비슷한 역할의 Prometheus·Loki 로 모니터링을 구성해 봤습니다.
87	모니터링에서 무엇을 보나요?	CPU, 메모리, 디스크, 네트워크 같은 자원 지표와, 응답 시간·에러율 같은 서비스 지표, 그리고 로그입니다. 기준을 넘으면 알림이 오게 합니다.

10. API·연동 기초

#	질문	30 초 답
88	API 란?	프로그램끼리 정해진 방식으로 기능이나 데이터를 주고받는 약속(인터페이스)입니다.
89	REST API 란?	자원을 URL 로 표현하고, HTTP 메서드(GET 조회, POST 생성, PUT·PATCH 수정, DELETE 삭제)로 다루며, JSON 으로 주고받는 API 설계 방식입니다.
90	HTTP 상태 코드의 종류는?	2xx 성공(200, 201), 3xx 이동, 4xx 요청 오류(400 잘못된 요청, 401 인증 필요, 403 권한 없음, 404 없음), 5xx 서버 오류(500, 502, 503)입니다.
91	JSON 이란?	키와 값으로 데이터를 표현하는 가벼운 텍스트 형식입니다. REST API 에서 가장 많이 씁니다.
92	SOAP 과 REST 의 차이는?	SOAP 은 XML 과 엄격한 규격(WSDL)을 쓰는 오래된 웹 서비스 방식으로 공공·금융 레거시에 많고, REST 는 HTTP 와 JSON 을 쓰는 가벼운 방식입니다. 온나라 전자결재 연동을 SOAP 으로 해 봤습니다.
93	JDBC 와 ODBC 의 차이는?	둘 다 프로그램이 DB 에 접속하는 표준 규격입니다. JDBC 는 Java 전용, ODBC 는 언어와 상관없이 쓰여서 Excel 이나 BI 도구가 많이 씁니다.
94	외부 API 를 연동할 때 무엇을 조심하나요?	타임아웃과 재시도, 인증 토큰 만료, 호출 제한, 응답이 명세와 다를 때의 예외 처리입니다. KLID 차량조회 API 의 오류 케이스를 분석해 보정 로직을 넣어 본 경험이 있습니다.

11. 태도·학습 (기술 면접에 섞여 나옴)

#	질문	30 초 답
95	가장 자신 있는 기술은? (후기)	SQL 실행 계획 기반 성능 튜닝입니다. 지방세 가상계좌 조회를 실행 계획부터 확인해 파티셔닝과 인덱스를 다시 설계하고 400~500ms 에서 50~60ms 로 줄였습니다. (꼬리질문 5 단계는 필수 정리 6 부)
96	새 기술을 계속 배워야 하는데 자신 있나요? (후기)	네. 비트교육센터, SQLD, SKALA 처럼 필요한 기술을 스스로 찾아 배워 왔고, 배운 것을 문서로 남기는 습관이 있습니다. 이번에도 Denodo 와 Cloudera 를 원리부터 공부하고 있습니다.
97	AI 를 어느 정도 활용하나요? (후기)	[직접 채우기: 실제로 쓰는 방식]. 서류에 있는 예: SKALA Community 에서 Claude API 로 실습 조교를 만들고 가드레일을 두어 시험함, 그룹웨어에 MCP 기반 학습 기록 기능 구현.
98	내가 써 본 기술을 우리 솔루션에 어떻게 접목하나요? (후기)	SQL 튜닝 경험은 Denodo 쿼리 성능 개선에, API 연동 경험은 Denodo 의 API 원천 연계에, Kubernetes 경험은 Denodo 컨테이너 배포와 ClovirCM 에, Kafka 경험은 Cloudera 수집 파이프라인에 쓸 수 있습니다.
99	모르는 질문을 받으면?	모른다고 솔직하게 말하고, 아는 데까지 말한 뒤, 어떻게 확인할지 말하겠습니다. 아는 척했다가 틀리면 고객의 신뢰를 잃는다고 생각합니다.
100	업무에 익숙해지는 데 얼마나 걸릴까요? (후기)	기본 설치와 연계 업무는 교육과 OJT 를 받으며 3~6 개월 안에 혼자 할 수 있게 하고, 1 년 안에 공식 자격을 따는 것이 목표입니다. 처음 맡는 일은 과정을 문서로 남겨 빨리 내 것으로 만들겠습니다.

12. Denodo·Cloudera 데이터 용어 정리

면접에서 이 단어들이 나왔을 때 **한 줄로 뜻을 말할 수 있으면** 충분합니다. 읽는 법은 [용어 읽는 법](용어_읽는법.md)을 따랐습니다. "(회사 웨비나)"는 굿모닝아이텍 유튜브 Cloudera 웨비나에서 설명한 표현입니다.

12-1. 데이터 가상화·Denodo 용어

Denodo 간단한 아키텍처

```
[소비자] BI 도구        애플리케이션      데이터 카탈로그
         (JDBC/ODBC)    (REST/OData)      (웹 화면)
              |              |               |
+-------------------------------------------------------------------+
| Denodo Platform                                                   |
|                                                                   |
|   +-------------------------------------------------------------+ |
|   | Virtual DataPort(VDP) 서버  = 핵심 질의 엔진, 데이터 저장 안 함 | |
|   |                                                             | |
|   | 접속/인증  ->  파서  ->  옵티마이저  ->  실행 엔진            | |
|   |          (뷰 펼치기)  (위임/조인 선택)  (조인/집계/스트림)    | |
|   |                                                             | |
|   | 캐시 모듈(외부 캐시 DB)   보안 모듈(권한, 행 제한)   메타데이터 | |
|   +-------------------------------------------------------------+ |
|                                                                   |
| 관리 도구: Design Studio(개발)  Solution Manager(배포)  Scheduler  |
+-------------------------------------------------------------------+
                          |  필터/조인은 원천으로 위임(푸시다운)
[원천]   Oracle    MSSQL    Snowflake    Hive/Impala    REST API
```

- **위(소비자):** BI 도구, 앱, 사용자가 Denodo 를 데이터베이스 하나처럼 접속합니다.
- **가운데(VDP):** 질의를 받아 원천별로 나눠 보내고 결과를 합칩니다. 데이터는 저장하지 않고, 캐시만 외부 캐시 DB 에 둡니다.
- **아래(원천):** 필터와 조인은 가능한 한 원천이 처리하게 넘겨서(푸시다운) 네트워크로 오는 데이터를 줄입니다.

Denodo 뷰 계층

```
+-------------------------------------------------------------+
| 소비 계층     보고서별, API 별로 필요한 모양과 권한          |
+-------------------------------------------------------------+
                    ^
+-------------------------------------------------------------+
| 비즈니스 계층 고객, 계좌, 거래처럼 업무 용어로 된 표준 모델  |
+-------------------------------------------------------------+
                    ^
+-------------------------------------------------------------+
| 통합 계층     코드와 이름 통일, 원천 간 조인                 |
```

```
+-------------------------------------------------------------+
                    ^
+-------------------------------------------------------------+
| 원천 계층      기본 뷰: 테이블이나 API 하나씩 그대로          |
+-------------------------------------------------------------+
                    ^
+-------------------------------------------------------------+
| 데이터 소스   Oracle, Snowflake, API 의 접속 정보            |
+-------------------------------------------------------------+
```

- 원천이 바뀌어도 영향이 아래 계층에서 끝나고, 비즈니스 계층을 여러 소비자가 재사용하도록 나눕니다.

용어

용어 (읽는 법)	한 줄 뜻
데이터 가상화	데이터를 옮기지 않고, 여러 원천을 하나의 가상 데이터베이스처럼 조회하게 하는 기술
논리적 데이터 패브릭	흩어진 데이터를 물리적으로 합치지 않고, 가상화·메타데이터·보안으로 하나처럼 엮는 구조. Denodo 가 자기 제품을 설명할 때 쓰는 말
논리적 데이터 웨어하우스	DW, 데이터 레이크, 업무 DB 를 그대로 두고 그 위에 가상화 계층을 얹어 하나의 분석 창구로 쓰는 구성
의미 계층 (시맨틱 레이어)	기술적인 테이블·컬럼을 '고객', '매출' 같은 업무 용어로 바꿔 보여 주는 계층. 사용자는 원천 구조를 몰라도 됨
원천 (데이터 소스)	실제 데이터가 있는 곳(Oracle, Snowflake, Hive, REST API, 파일). Denodo 에서는 접속 정보를 담은 객체 이름이기도 함
VDP (Virtual DataPort, 브이디피)	Denodo 의 핵심 질의 엔진 서버. 질의를 받아 원천에 나눠 보내고 결과를 합침
VQL (브이큐엘)	Denodo 의 질의·정의 언어. 조회는 SQL 과 거의 같음
기본 뷰 (base view)	원천의 테이블이나 API 응답 하나를 Denodo 로 가져온 뷰
파생 뷰 (derived view)	기본 뷰를 조인·필터·유니온·집계해서 만든 뷰
인터페이스 뷰	소비자에게 보이는 모양만 정하고 실제 구현은 바꿀 수 있게 한 뷰. 원천을 바꿔도 쿼리는 그대로
평탄화 (flatten, 플래튼)	JSON 처럼 중첩된 데이터나 배열을 표 형태로 펼치는 것
검색 방법 (search method)	API 같은 원천이 어떤 조건을 꼭 받아야 조회되는지 정의한 것
위임 / 푸시다운 (delegation, pushdown)	필터·조인·집계를 원천 DB 가 처리하게 넘기는 것. 네트워크로 오는 데이터를 줄임
브랜치 프루닝	결과에 필요 없는 조인이나 유니온 가지를 실행 전에 잘라 내는 최적화
데이터 이동 (data movement)	작은 테이블을 큰 테이블이 있는 DB 로 임시 복사해 조인을 그 DB 에서 하게 하는 최적화

NESTED 조인	한쪽 결과의 키 값으로 다른 원천에 조건 질의를 보내는 조인. API 원천과 조인할 때 씀
비용 기반 최적화 (CBO)	통계로 행 수를 추정해 가장 싼 실행 방법을 고르는 것. 통계가 없으면 잘못 고름
뷰 통계	뷰의 행 수, 값 분포 같은 정보. 옵티마이저 판단의 근거
실행 트레이스	질의가 실행된 뒤 단계별 시간·행 수·원천에 보낸 SQL 을 보여 주는 기록. 성능 분석의 출발점
캐시 / 캐시 DB	자주 쓰는 결과를 외부 DB 에 저장해 두고 재사용하는 것. 부분 캐시(이전 질의 결과 재사용)와 전체 캐시(뷰 전체 적재)가 있음
요약 (Summaries, 서머리즈)	미리 만든 집계 결과를 옵티마이저가 알아서 대신 쓰게 하는 기능
MPP 가속	대량 연산을 병렬 처리 엔진에 넘겨 빠르게 처리하는 것
행 제한 / 마스킹	역할에 따라 볼 수 있는 행을 제한하는 것 / 주민번호처럼 민감한 값을 가리는 것
패스스루 (pass-through)	서비스 계정 대신 사용자 본인 계정으로 원천에 접속해 원천의 권한을 그대로 쓰는 방식
Data Catalog	업무 사용자가 뷰를 검색하고 설명·태그·출처를 보는 화면
Solution Manager	개발·검증·운영 환경을 관리하고 뷰 정의를 다음 환경으로 배포(승격)하는 도구
환경 승격 (프로모션)	개발에서 만든 뷰를 검증, 운영 환경으로 차례로 옮기는 것

12-2. Cloudera·Hadoop 용어

Cloudera 간단한 아키텍처

```
[원천] 업무 DB, 로그, 파일, IoT
          |
[수집] NiFi(흐름 수집)  Kafka(실시간 메시징)
          |
+-------------------------------------------------------------------------+
| Cloudera (CDP Private Cloud Base)                                       |
|                                                                         |
| 처리/조회  Spark(가공, 스트리밍)  Hive(배치 SQL)  Impala(대화형 SQL)   |
| 자원 관리  YARN(CPU, 메모리 배분)  ZooKeeper(조정)                      |
| 저장       HDFS(128MB 블록, 3벌 복제)  Ozone  Kudu  HBase              |
| 테이블     Hive 메타스토어(정의 공유)  Iceberg, Parquet, ORC            |
| 보안(SDX)  Ranger(권한)  Atlas(리니지)  Kerberos(인증)  Knox           |
| 관리       Cloudera Manager(설치, 설정, 모니터링, 재시작)               |
+-------------------------------------------------------------------------+
          |
[소비] BI 도구(Impala)  Denodo(JDBC 로 연결)  분석가(Hue)  Cloudera AI
```

- **흐름:** 원천 → 수집(NiFi, Kafka) → 저장(HDFS 등) → 가공·조회(Spark, Hive, Impala) → 소비(BI, Denodo)
- **SDX** 가 모든 층에 걸쳐 권한·리니지·인증을 맡고, **Cloudera Manager** 가 전체 서비스를 설치하고 관리합니다.

Cloudera 노드 배치

```
+-------------------------------------------------------------+
| 마스터 노드 x3   NameNode(Active/Standby), ResourceManager,    |
|                  ZooKeeper, JournalNode, Hive 메타스토어       |
+-------------------------------------------------------------+
        |  관리, 감시(하트비트)
+-------------------------------------------------------------+
| 워커 노드 xN     DataNode + NodeManager + Impala 데몬          |
|                  (늘릴수록 저장 용량과 처리 능력이 함께 증가)  |
+-------------------------------------------------------------+
+-------------------------------------------------------------+
| 기타             게이트웨이(사용자 접속, Hue), Cloudera Manager 서버 |
+-------------------------------------------------------------+
```

- 마스터를 3 대로 두는 이유는 ZooKeeper·JournalNode 가 과반 합의를 쓰기 때문입니다(3 대면 1 대 장애를 견딤).
- 워커를 늘리면 저장 용량과 처리 능력이 함께 늘어납니다(수평 확장).

용어

용어 (읽는 법)	한 줄 뜻
CDP (씨디피)	Cloudera 와 Hortonworks 가 합쳐 만든 통합 데이터 플랫폼
Private Cloud Base	고객 데이터센터의 서버·VM 에 설치하는 전통적인 Cloudera 클러스터. 국내 구축의 대부분
Data Services	데이터 웨어하우스, 데이터 엔지니어링, AI 같은 서비스를 쿠버네티스 위 컨테이너로 운영하는 형태
Cloudera Manager (씨엠)	클러스터의 서비스 설치·설정·모니터링·재시작을 한 화면에서 하는 관리 도구
파슬 (parcel)	Cloudera Manager 가 배포하는 소프트웨어 묶음. 버전을 통째로 바꾸고 되돌리기 쉬움
클러스터	여러 서버를 묶어 하나처럼 쓰는 구성
마스터 / 워커 / 게이트웨이 노드	관리 역할(NameNode 등) / 저장·처리 역할(DataNode 등) / 사용자 접속 지점
HDFS (에이치디에프에스)	파일을 블록으로 쪼개 여러 서버에 복제 저장하는 분산 파일 시스템
블록	HDFS 가 파일을 나누는 단위. 기본 128MB

복제 계수 (replication factor)	블록을 몇 벌 저장할지. 기본 3
랙 인식 (rack awareness)	복제본을 서로 다른 랙(서버 선반)에 나눠 둬서 랙 장애에도 데이터를 지키는 것
NameNode / DataNode	파일·블록 위치를 관리하는 관리자 / 실제 블록을 저장하는 일꾼
Secondary NameNode	메타데이터 파일을 주기적으로 정리하는 도우미. **대기 서버가 아님**
JournalNode / ZooKeeper / ZKFC	NameNode HA 를 위한 변경 기록 저장소 / 분산 조정 서비스 / 장애 감지와 자동 전환 담당
작은 파일 문제	작은 파일이 너무 많으면 NameNode 메모리가 부족해지는 HDFS 의 대표 한계
디커미션 (decommission) / 밸런서	노드를 안전하게 빼는 작업 / 노드 간 데이터 양을 고르게 맞추는 도구
YARN (얀)	클러스터의 CPU·메모리를 작업에 나눠 주는 자원 관리자
큐 (queue)	YARN 에서 부서·업무별로 자원 몫을 나눠 둔 단위
MapReduce	map → shuffle → reduce 로 처리하는 초기 Hadoop 처리 방식. 디스크를 많이 써서 느림
Hive / Hive 메타스토어 / HiveServer2	SQL 배치 엔진 / 테이블 정의 저장소(Spark·Impala 도 공유) / JDBC 접속 서버
Impala (임팔라)	대화형 SQL 엔진. BI 도구의 빠른 조회용
Spark / 드라이버 / 익스큐터	메모리 기반 분산 처리 엔진 / 작업 계획을 세우는 프로세스 / 실제로 일하는 프로세스
셔플	데이터를 키별로 다시 나눠 서버끼리 주고받는 과정. 가장 비싼 단계
파티션	데이터를 나눈 단위. Hive 에서는 날짜별 디렉터리, Spark 에서는 병렬 처리 단위, Kafka 에서는 토픽을 나눈 단위
Kafka 토픽 / 오프셋 / 컨슈머 그룹	메시지 분류 이름 / 어디까지 읽었는지 위치 / 메시지를 나눠 읽는 소비자 묶음
NiFi (나이파이)	화면에서 흐름을 그려 데이터를 수집·변환·전달하는 도구
Ozone (오존)	S3 와 호환되는 오브젝트 스토리지. HDFS 의 작은 파일 한계를 보완
Kudu (쿠두)	빠른 분석과 행 단위 갱신을 함께 하는 저장소. 경합이 많은 갱신용 (회사 웨비나)
HBase (에이치베이스)	키로 대량의 데이터를 빠르게 읽고 쓰는 NoSQL DB
Parquet / ORC (파케이 / 오알씨)	분석에 유리한 열 기반 파일 포맷. 필요한 컬럼만 읽고 압축이 잘 됨
Iceberg (아이스버그)	파일 묶음을 테이블처럼 관리하는 테이블 포맷. 스냅샷으로 읽기·쓰기 충돌을 피하고, 데이터를 다시 쓰지 않고 파티션을 바꾸고(파티션 진화), 과거 시점을 조회(타임 트래블)함 (회사 웨비나)
데이터 레이크하우스	데이터 레이크에 Iceberg 같은 테이블 포맷을 더해 DW 처럼 트랜잭션과 SQL 을 쓰는 구조

스키마 온 리드 / 스키마 온 라이트	읽을 때 구조를 정함(레이크) / 저장할 때 구조를 정함(DW)
SDX (에스디엑스)	Cloudera 의 공통 보안·거버넌스 계층. 프록시가 아니라 시스템 사이의 "컨텍스트 레이어" (회사 웨비나)
Ranger (레인저)	누가 어떤 데이터에 접근할 수 있는지 정책으로 관리하고 감사 기록을 남김
Atlas (아틀라스)	메타데이터, 리니지, 분류 태그를 관리
리니지	데이터가 어디서 와서 어디를 거쳐 어떻게 가공됐는지 보여 주는 이력 (회사 웨비나)
분류 태그	데이터에 '개인정보' 같은 꼬리표를 붙여 검색과 보안 정책에 쓰는 것
Knox (녹스)	클러스터 외부 접속의 관문(게이트웨이), SSO 연동
Kerberos / KDC / principal / keytab	티켓 기반 인증 / 티켓 발급 서버 / 사용자·서비스의 인증 이름 / 서비스용 키 파일
Cloudera AI (옛 CML·CDSW)	데이터 과학자가 모델을 만들고 배포·관리하는 작업 공간. 회사가 AI 전략의 중심으로 삼은 제품

12-3. 함께 알아 둘 데이터 일반 용어

용어	한 줄 뜻
배치 / 스트리밍	모아서 한꺼번에 처리 / 들어오는 대로 실시간 처리
CDC (씨디씨)	원천 DB 의 변경분만 잡아서 전달하는 방식(로그 기반이 많음)
데이터 마트	부서·주제별로 잘라 놓은 작은 DW
팩트 / 차원	측정값(매출액, 수량) 테이블 / 설명 정보(고객, 상품, 날짜) 테이블. 둘을 별 모양으로 묶은 것이 스타 스키마
마스터 데이터 (MDM)	고객·상품처럼 여러 시스템이 함께 쓰는 핵심 기준 데이터와 그 관리
데이터 품질	데이터가 정확하고, 빠짐없고, 최신이고, 일관된 정도
데이터 사일로	부서·시스템마다 데이터가 따로 갇혀 서로 연결되지 않는 상태. 데이터 가상화와 SDX 가 풀려는 문제

13. 공고 필수역량별 예상 질문

공고 원문(사람인 rec_idx 55473228)의 필수역량·우대사항 문장을 그대로 제목으로 두고, 면접관이 그 문장을 보며 물을 질문을 정리했습니다. 이런 질문은 대개 **"경험 있나요?" → "어떻게 했나요?" → "왜 그렇게 했나요?"** 순서로 들어옵니다. 그래서 답도 **경험(사실) → 원리 → 이 직무에서 어떻게 쓸지** 순서로 짰습니다. 경험이 없는 부분은 솔직하게 없다고 하고, 공부한 원리로 답합니다.

13-1. [공통 필수] Linux 에서 프로세스·권한·리소스 관리, Shell Script 작성 경험

#	질문	30 초 답
101	Linux 서버를 다뤄 본 경험은?	개인 프로젝트에서 Linux VM 에 kubeadm 으로 Kubernetes 클러스터를 직접 구성하고, AWS EC2 에 서비스를 배포해 봤습니다. SKALA 에서는 서비스를 컨테이너로 배포·운영했습니다. [직접 채우기: SI 운영에서 Linux 서버를 다룬 경험이 있다면 한 문장]
102	프로세스는 어떻게 확인하고 종료하나요?	`ps -ef` 나 `top` 으로 PID 와 자원 사용량을 보고, `kill`(SIGTERM)로 정상 종료합니다. 정리 작업 없이 끝나는 `kill -9` 는 데이터나 락 파일이 꼬일 수 있어서 마지막 수단으로만 씁니다. 운영 서비스는 `systemctl stop` 으로 내립니다.
103	솔루션 계정과 권한은 어떻게 관리하나요?	솔루션은 root 가 아니라 전용 서비스 계정으로 실행하고, 설치·로그·데이터 디렉터리의 소유자를 그 계정으로 맞춥니다. 실행 파일은 755, 설정 파일은 640 처럼 필요한 만큼만 권한을 줍니다. 최소 권한 원칙입니다.
104	서버 리소스는 어떻게 확인하나요?	CPU 는 `top` 과 load average 를 코어 수와 비교하고, 메모리는 `free -h` 의 available, 디스크는 `df -h` 와 `du` 로 봅니다. I/O 대기가 높으면 `iostat` 으로 디스크를 봅니다.
105	"Too many open files" 에러가 나면?	프로세스가 열 수 있는 파일 수 한도(ulimit nofile)를 넘은 것입니다. `/proc/PID/limits` 로 실제 한도를 보고, systemd 서비스면 유닛 파일의 `LimitNOFILE` 로 올립니다. 연결이 계속 늘기만 하면 누수이니 애플리케이션을 고쳐야 합니다.
106	free 메모리가 거의 없는데 괜찮은가요?	Linux 는 남는 메모리를 파일 캐시로 쓰기 때문에 free 가 작아도 정상일 수 있습니다. available 이 충분한지, 스왑이 계속 늘지 않는지를 봅니다.
107	Shell Script 를 작성해 본 경험은?	개인 프로젝트에서 kubeadm VM 클러스터를 한 번에 구성하는 배포 스크립트를 작성했습니다. [직접 채우기: 스크립트가 자동화한 단계와, 실패했을 때 어떻게 처리했는지]

108	Shell Script 를 쓸 때 무엇을 주의하나요?	`set -euo pipefail` 로 실패를 그냥 넘기지 않게 하고, 변수는 항상 큰따옴표로 감쌉니다. 종료 코드로 성공·실패를 알리고, 실행 기록을 로그로 남깁니다. 지우는 명령은 대상 경로를 먼저 확인합니다.
109	매일 새벽에 도는 작업은 어떻게 만드나요?	cron 에 등록합니다. cron 은 환경 변수가 거의 없어서 명령을 절대 경로로 쓰고, 출력을 로그 파일로 남겨 실패를 확인할 수 있게 합니다. 작업 간 순서나 재시도가 복잡하면 Airflow 같은 도구를 씁니다.
110	로그에서 에러가 몇 번 났는지 시간대별로 보려면?	`grep "ERROR" app.log` 로 에러 줄만 뽑고, `awk` 로 시간 부분을 잘라 `sort | uniq -c` 로 셉니다. 한 줄 명령으로 빠르게 장애 시점을 찾을 수 있습니다.
111	솔루션을 systemd 서비스로 등록하는 이유는?	서버가 재부팅돼도 자동으로 뜨고, 비정상 종료 때 자동 재시작되며, 로그를 journalctl 로 한곳에서 볼 수 있기 때문입니다. 실행 계정과 파일 한도도 유닛 파일에서 정합니다.

13-2. [Denodo 필수] SQL 기반 데이터 조회·처리와 성능 최적화 능력

#	질문	30 초 답
112	SQL 성능을 최적화해 본 경험은?	지방세 가상계좌 시스템에서 발급 요청이 몰리면 조회가 400~500ms 까지 늘었습니다. 실행계획으로 대량 조회 때 디스크 I/O 가 몰리는 구간을 찾아, 은행코드 기준 리스트 파티셔닝, 복합 인덱스 재설계, Parallel Query 힌트로 50~60ms 까지 약 85% 줄였습니다.
113	실행 계획에서 무엇을 보나요?	테이블을 전체로 읽는지 인덱스로 읽는지, 조인 순서와 방식, 그리고 예상 행 수와 실제 행 수의 차이를 봅니다. 차이가 크면 통계가 틀린 것이라 옵티마이저가 잘못된 계획을 고릅니다. [직접 채우기: 가상계좌 튜닝 때 실제로 본 부분]
114	인덱스가 있는데 안 타는 이유는?	조건 컬럼을 함수로 가공했거나, 타입이 달라 형변환이 일어났거나, LIKE 를 앞에 %를 붙여 썼거나, 복합 인덱스의 선두 컬럼이 조건에 없을 때입니다. 조건에 맞는 행이 너무 많으면 옵티마이저가 일부러 전체 읽기를 고르기도 합니다.
115	조인 방식에는 무엇이 있나요?	한쪽이 작고 다른 쪽에 인덱스가 있을 때 좋은 중첩 루프, 큰 데이터끼리 등호 조인할 때 좋은 해시 조인, 정렬된 데이터를 맞춰 가는 정렬 병합 조인이 있습니다. Denodo 도 원천이 여러 곳일 때 같은 원리로 조인 방식을 고릅니다.
116	파티셔닝은 언제 쓰나요?	큰 테이블을 날짜나 코드 같은 기준으로 나눠, 조건에 맞는 파티션만 읽게 할 때 씁니다. 가상계좌는 조회 조건이 은행코드였고

		은행코드는 정해진 값 목록이라 리스트 파티셔닝을 썼습니다.
117	동시 요청이 몰릴 때 생긴 문제는 어떻게 풀었나요?	가상계좌 발급 요청이 몰리면 같은 계좌를 가져가려는 경합이 생겼는데, FOR UPDATE SKIP LOCKED 로 이미 잠긴 행은 건너뛰고 비어 있는 다른 계좌를 바로 가져가게 해서 대기를 없앴습니다.
118	그룹별 최신 1 건만 뽑으려면?	`ROW_NUMBER() OVER (PARTITION BY 그룹 ORDER BY 일시 DESC)`로 순번을 매긴 뒤 순번이 1 인 행만 고릅니다. 자기 자신과 조인하는 것보다 간단하고 빠릅니다.
119	데이터 가상화에서 SQL 최적화 능력이 왜 필요할까요?	Denodo 는 받은 질의를 원천 DB 에 나눠 보내기 때문에, 결국 성능은 원천에 가는 SQL 과 조인 방식에 달려 있습니다. 필터가 원천에 위임됐는지, 원천 쿼리가 인덱스를 타는지 볼 수 있어야 느린 원인을 찾을 수 있습니다.

13-3. [Denodo 필수] Python 또는 Java 로 REST API 를 연동한 경험

#	질문	30 초 답
120	외부 API 를 연동해 본 경험은?	전기차 충전구역 단속 시스템에서 한국지역정보개발원 차량조회 API 를 연동했고, 오류 케이스를 분석해 보정 로직을 넣어 단속 데이터 정확도를 95% 이상으로 맞췄습니다. 온나라 전자결재는 SOAP 으로 연동했고, 개인 프로젝트에서는 Spring Boot 로 REST API 를 직접 설계했습니다.
121	Java 로 REST API 를 호출하는 코드는 어떻게 짜나요?	HTTP 클라이언트로 요청을 만들고, 연결·응답 타임아웃을 정하고, 상태 코드를 확인한 뒤 JSON 을 객체로 바꿉니다. 실패하면 예외를 던지고 로그를 남깁니다. Python 이면 requests 로 같은 흐름입니다.
122	API 인증 방식에는 무엇이 있나요?	API 키를 헤더에 넣는 방식, 서버끼리 쓰는 OAuth2 client credentials 방식, 서명된 토큰을 쓰는 JWT 방식이 대표적입니다. 개인 프로젝트에서 세션 인증을 JWT(RS256)로 바꿔 본 경험이 있습니다.
123	API 호출이 실패하면 어떻게 하나요?	일시적인 오류(타임아웃, 5xx, 429)는 간격을 늘려 가며 재시도하고, 같은 요청이 두 번 처리돼도 문제가 없는지 확인합니다. 4xx 처럼 요청 자체가 잘못된 경우는 재시도하지 않고 원인을 고칩니다.
124	데이터가 많아 API 가 나눠서 준다면?	페이지네이션입니다. offset·limit 이나 다음 페이지 커서를 받아 끝까지 반복해서 가져오고, 중간에 실패하면 어디까지 받았는지 기록해 이어받습니다.
125	API 응답이 명세와 다르게 오면?	차량조회 API 에서 실제로 겪었습니다. 오류 케이스를 모아

		패턴을 분석하고, 빈 값이나 형식이 다른 값을 보정하는 로직을 넣었습니다. 연동에서는 정상 응답보다 예외 응답을 어떻게 다루느냐가 데이터 품질을 좌우합니다.
126	Denodo 에서 API 를 원천으로 쓸 때 주의할 점은?	API 는 DB 처럼 조건 질의를 자유롭게 받지 못해서, 어떤 값을 꼭 줘야 하는지(검색 방법)를 정의하고, 다른 원천의 결과로 필요한 만큼만 호출(NESTED 조인)하게 합니다. 호출 제한, 토큰 만료, 응답 시간도 확인합니다.

13-4. [Cloudera 필수] Hadoop Eco-System(HDFS, Hive, Spark 등)에 대한 이해

#	질문	30 초 답
127	Hadoop 생태계를 설명해 보세요.	HDFS 가 데이터를 여러 서버에 나눠 저장하고, YARN 이 자원을 나눠 주며, 그 위에서 Hive·Impala 가 SQL 로 조회하고 Spark 가 가공합니다. Kafka·NiFi 로 수집하고, Ranger·Atlas·Kerberos 로 보안과 거버넌스를 맡습니다. Cloudera 는 이를 Cloudera Manager 로 묶어 관리합니다.
128	HDFS 구조와 장애 대응은?	NameNode 가 메타데이터를, DataNode 가 128MB 블록을 3 벌씩 저장합니다. DataNode 가 죽으면 다른 노드의 복제본으로 다시 3 벌을 맞추고, NameNode 는 Active·Standby 로 HA 를 구성합니다.
129	Hive, Impala, Spark 는 언제 쓰나요?	Hive 는 대량 배치 SQL, Impala 는 BI 도구의 빠른 대화형 조회, Spark 는 프로그램으로 하는 복잡한 가공과 스트리밍입니다. 셋 다 Hive 메타스토어의 같은 테이블 정의를 씁니다.
130	분산 처리 작업이 느리면 어디를 보나요? (담당업무: 성능 분석·튜닝)	Spark 라면 어느 스테이지에서 시간이 걸리는지 보고, 셔플이 큰지, 특정 태스크만 오래 걸리는 데이터 쏠림이 있는지 확인합니다. YARN 에서 자원을 충분히 받는지, 작은 파일이 너무 많지 않은지도 봅니다.
131	Spark 튜닝 기본은?	셔플을 줄이는 것이 핵심입니다. 작은 테이블은 브로드캐스트 조인으로 셔플을 피하고, 셔플 파티션 수를 데이터 크기에 맞추고, 쏠린 키는 나눠 처리합니다. 파일은 Parquet 같은 열 기반 포맷을 씁니다.
132	Hadoop 을 직접 써 봤나요?	실무로 운영해 본 적은 없습니다. 대신 Kafka 로 메시지를 받아 저장하는 파이프라인과 Helm 기반 Airflow 배포를 SKALA 에서 다뤄 봤고, 지금은 HDFS·Hive·Spark 구조와 Cloudera 운영을 원리부터 공부하고 있습니다.

13-5. [Cloudera 필수] 네트워크, DNS, 방화벽 등 인프라 기본 지식

#	질문	30 초 답
133	클러스터 설치 전에 네트워크에서 무엇을 확인하나요?	모든 노드의 호스트 이름과 DNS 정방향·역방향 해석이 일치하는지, 시간 동기화가 되는지, 노드 사이와 사용자 구간의 필요한 포트가 방화벽에서 열려 있는지, 대역폭이 충분한지 확인합니다.
134	DNS 정방향·역방향이 왜 중요한가요?	Hadoop 노드는 서로를 호스트 이름(FQDN)으로 부르고, Kerberos 인증 이름에도 호스트 이름이 들어가서, 이름과 IP 가 양방향으로 일치하지 않으면 인증이 실패하거나 노드가 잘못 등록됩니다.
135	방화벽 오픈은 어떻게 요청하나요?	출발지 IP, 목적지 IP, 포트, 프로토콜을 표로 정리해 고객사 보안 담당자에게 요청합니다. 서버 자체 방화벽과 네트워크 구간 방화벽이 따로 있어서 둘 다 확인해야 합니다.
136	연결이 안 될 때 refused 와 timeout 은 어떻게 다른가요?	refused 는 상대 서버에는 닿았는데 그 포트에서 서비스가 안 떠 있거나 거부한 것이고, timeout 은 중간에서 패킷이 버려진 것이라 방화벽이나 라우팅 문제입니다. `nc -zv 호스트 포트`로 구분합니다.
137	192.168.10.77/26 의 네트워크 주소는?	/26 은 64 개 단위라 64~127 구간에 속합니다. 네트워크 주소는 192.168.10.64, 브로드캐스트는 .127, 쓸 수 있는 호스트는 .65~.126 입니다.
138	클러스터에서 시간 동기화가 왜 필요한가요?	로그 순서를 맞추고, 특히 Kerberos 인증은 서버 간 시계가 기본 5 분 이상 어긋나면 실패하기 때문입니다. 모든 노드에 NTP(chrony)를 설정합니다.

13-6. [공통 우대] Docker/K8s 기반 컨테이너 환경 구축 또는 운영 경험

#	질문	30 초 답
139	컨테이너 환경을 구축·운영해 본 경험은?	SKALA 에서 Spring Boot 서비스 4 개를 k3d 클러스터에 Helm 과 Argo CD 로 배포해 운영했고, 5 일 동안 요청 7 만 1 천여 건을 서버 오류 없이 처리했습니다. AutoTC 에서는 Jenkins 와 Argo CD, Argo Rollouts 로 블루그린 배포를 구성했고, 개인 프로젝트에서는 kubeadm 으로 클러스터를 직접 만들었습니다.
140	컨테이너가 메모리 한도를	커널이 그 컨테이너를 강제 종료하고 OOMKilled 상태가 됩니다.

	넘으면?	SKALA 에서 부하 시험과 `kubectl top` 으로 메모리 한도에 가까운 서비스를 찾아 한도를 조정한 경험이 있습니다.
141	Helm 과 Argo CD 는 무엇인가요?	Helm 은 쿠버네티스 배포 설정을 템플릿과 값으로 묶어 재사용하는 패키지 도구이고, Argo CD 는 Git 에 적힌 상태대로 클러스터를 자동으로 맞춰 주는 GitOps 도구입니다.
142	컨테이너 경험을 이 직무에 어떻게 쓰나요?	Denodo 도 컨테이너 배포를 지원하고, Cloudera 의 데이터 서비스도 쿠버네티스 위에서 돕니다. 회사 제품 ClovirCM 도 컨테이너 관리 포털이라, 고객 환경에 데이터 플랫폼을 컨테이너로 올리고 운영하는 데 바로 쓸 수 있습니다.

13-7. [Denodo 우대] 논리적·물리적 데이터 모델링 경험

#	질문	30 초 답
143	논리 모델링과 물리 모델링의 차이는?	논리 모델링은 업무 관점에서 엔터티, 속성, 관계를 정의하는 것이고, 물리 모델링은 특정 DB 에 맞게 테이블, 컬럼 타입, 인덱스, 파티션을 정하는 것입니다.
144	데이터 모델링을 해 본 경험은?	그룹웨어 프로젝트 1 차에서 ERD 와 API 명세를 작성했고, 2 차에서는 서비스를 나누면서 MySQL 스키마도 서비스별로 분리했습니다. [직접 채우기: 모델링하며 고민한 점 한 가지]
145	반정규화는 언제 하나요?	조인이 너무 많아 조회가 느리고, 데이터 변경보다 조회가 훨씬 잦을 때 일부러 중복을 허용합니다. 대신 데이터 일관성을 지킬 방법을 함께 정합니다.
146	Denodo 에서 모델링이 왜 중요한가요?	여러 원천을 묶은 비즈니스 계층이 곧 고객이 쓰는 표준 데이터 모델이기 때문입니다. 고객·계좌·거래처럼 업무 용어로 모델을 잘 세워야 여러 부서가 같은 뷰를 재사용하고, 원천이 바뀌어도 영향이 적습니다.

13-8. [Cloudera 우대] Airflow, Kafka, NiFi 등을 활용한 데이터 파이프라인 경험

#	질문	30 초 답
147	Kafka 를 써 본 경험은?	SKALA 커뮤니티 서비스에서 채팅 메시지를 Kafka 를 거쳐 저장했습니다. 방 번호를 키로 써서 같은 방의 메시지 순서를 지켰고, 저장에 실패하면 두 번 재시도한 뒤 DLT 로 보관해 유실을 막았습니다.
148	Airflow 는 무엇이고 써 봤나요?	작업 순서와 의존 관계를 DAG 로 정의해 스케줄링하고, 실패 시 재시도하는 워크플로 도구입니다. SKALA 에서 Helm 으로 Airflow 를

		배포하고 데이터 파이프라인 설계를 배웠습니다. [직접 채우기: 직접 만든 DAG 가 있다면]
149	Kafka 와 NiFi 는 어떻게 다른가요?	Kafka 는 대량의 이벤트를 받아 보관하고 여러 소비자에게 전달하는 메시징 플랫폼이고, NiFi 는 여러 원천의 데이터를 화면에서 흐름으로 그려 수집·변환·라우팅하는 도구입니다. 함께 쓰면 NiFi 로 모아 Kafka 로 흘립니다.
150	파이프라인에서 데이터 유실과 중복은 어떻게 막나요?	실패하면 재시도하되, 같은 데이터가 두 번 들어와도 결과가 같도록 키 기준으로 처리하고, 끝내 실패한 데이터는 따로 보관해 원인을 분석합니다. SKALA 에서 재시도 후 DLT 로 보낸 것이 그 예입니다.

연습 방법

1. **소리 내어 답하기:** 표의 답을 외우지 말고, 첫 문장(정의)만 기억한 뒤 자기 말로 30 초 안에 답해 봅니다.

2. **(후기) 표시부터:** 실제로 나온 질문(20 여 개)을 먼저 완벽하게 합니다.

3. **Linux 명령어는 손으로:** 4-1 표의 명령을 실제로 쳐 봅니다. 후기에 "답이 부족하면 테스트를 봤다"는 내용이 있습니다.

4. **채팅으로 점검:** "기초 질문 모의 면접"이라고 보내면 이 100 개에서 무작위로 물어보고 채점합니다.