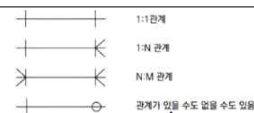


정보처리 실기 정리 / 약술형 대비

DBMS란?	다수의 사용자들이 데이터베이스 내의 데이터를 접근할 수 있는 기능을 지원하는 소프트웨어 도구의 집합이다.			
DBMS 무결성의 종류③	개체 무결성	기본키를 구성하는 속성은 NULL이나 중복값을 가질 수 없다.		
	참조 무결성	외래키 값은 NULL이거나 참조 릴레이션의 기본키 값과 동일해야 한다. 릴레이션은 참조할 수 없는 외래키 값을 가질 수 없다.		
	도메인 무결성	특정 속성의 값이, 그 속성이 정의된 도메인에 속한 값이어야 한다.		
네트워크 클래스란? Classful Network	네트워크 단말의 증가로 가용 가능한 IPv4의 주소가 부족 사용 목적에 따라 IP의 대역대를 나누어 관리하는 것이다. (첫째 옥텟을 앞부터 0, 1로 나눔)			
네트워크 클래스의 종류 A~E	A클래스	대륙 안 통신	D클래스	그룹 통신
	B클래스	국가 안 통신	E클래스	연구용 통신
	C클래스	기업 내 통신		
RAID란? Redundant Array of Inexpensive/Independent Disk	하스디스크의 장애로 인한 데이터 손실을 방지하기 위한 기술로, 여러 개의 디스크를 배열하여 속도와 안정성의 증대 및 효율성, 가용성의 증대를 목적으로 한다.			
형상관리란? SCM Software Configuration Management	소프트웨어 개발 프로세스 각 단계에서 소프트웨어의 변경을 추적하고 관리하는 일련의 활동			
형상관리 툴 종류③	CVS	Concurrent Version System 가장 오래된 형상관리 도구(1990) 중 하나로, 서버는 단순한 명령 구조를 가진 장점이 있고, 텍스트 기반 코드만 지원하는 단점이 있다.		
	SVN	Subversion CVS의 단점을 보완, GUI도구 존재하고 압축을 통해 서버의 공간을 절약		
	Git	리눅스 커널 개발을 위해 만들어짐. CVS와 SVN의 단점을 모두 보완하는 장점이 있고 중앙 집중형이 아닌 분산형 방식으로 스스로 저장공간이 필요하다.		
데이터베이스, 개념적 설계의 구성요소 ③	엔티티 Entity	사물 또는 사건, 개체라고도 한다.	ERD에서 사각형	
	속성 Attribute	엔티티가 가지고 있는 요소, 성질	ERD에서 동그라미, 표	
	관계 Relationship	두 엔티티 간의 관계를 정의한다.		
이상현상 Anomaly	데이터 중복성에 의해서, 릴레이션 조작 시 예기치 못한 현상, 데이터 불일치 현상 삽입이상 / 삭제이상/ 갱신이상			
DB 인덱스란?	데이터 레코드를 빠르게 접근하기 위해 “키 값, 포인터” 쌍으로 구성되는 데이터 구조 (클러스터드 인덱스-정렬저장 / 논클러스터드 인덱스)			
클러스터란? Cluster	데이터 액세스 효율을 향상시키기 위해, 동일한 성격의 데이터를 동일한 데이터 블록에 저장 클러스터의 분포도가 넓을수록 유리하다.			
파티션이란? Partition	대용량의 테이블이나 인덱스를 작은 논리적 단위인 파티션으로 나눈 것 (범위 분할 / 해시 분할 / 리스트 분할 / 조합Composite 분할)			

IDE란?(통합개발 환경) Integrated Development Environment	개발에 필요한 다양한 틀을, 하나의 인터페이스로 통합하여 제공한다.		
WAS란? Web Application Server	HTTP를 통해 애플리케이션을 수행해주는 <u>미들웨어</u> DB 조회 및 다양한 로직 처리 요구 시, 동적인 콘텐츠를 제공하기 위해 만들어진 애플리케이션 서버		
EAI란? Enterprise Application Integration	기업에서 운영하는 서로 다른 플랫폼 및 애플리케이션 간의 정보를 전달, 연계, 통합 해주는 솔루션 // 미들웨어를 이용하여 <u>비즈니스 로직을</u> 중심으로 기업 내 애플리케이션을 통합 연계		
EAI 구축 유형④	포인트 투 포인트	Point-to-point	가장 기초적인 애플리케이션 통합방법, 1:1
	허브 앤 스포크	Hub & Spoke	단일한 점점의 허브 시스템을 통하여 데이터를 전송 중앙 집중 방식
	메시지 버스	Message Bus	애플리케이션 사이에 미들웨어, 미들웨어 통합 방식
	하이브리드	Hybrid	그룹 내 Hub&Spoke, 그룹 간=Message Bus 통합 방식
ESB란? Enterprise Service Bus	기업에서 운영되는 서로 다른 플랫폼 및 애플리케이션 간을 하나의 시스템으로 느슨하게 결합함 // 미들웨어를 이용해 <u>서비스 중심</u> 으로 서비스를 지원하기 위한 관련 시스템과 유기적 연계 버스 방식의 분산형 토폴로지 구성		
웹 서비스 Web Service 표준 기술③	SOAP	Simple Object Access Protocol	단순 객체 접근 프로토콜 /프로토콜 , MS사 HTTP, HTTPS, SMTP 사용 + XML기반 메시지 교환
	WSDL	Web Service Description Language	웹 서비스 기술 언어 /설명서 웹 서비스에 대한 상세정보 기술, XML기반 제공
	UDDI	Universal Description Discovery and Integration	전역 비즈니스 레지스트리 /저장소 WSDL+서비스 내용을 인터넷 상에 등록, XML기반 저장소
IPC란? Inter-Process Communication	운영체제에서 <u>프로세스 간</u> 데이터를 주고받기 위한 통신 기술		
암호화 전송 보안 기술④	IPSec	IP Security	// IP계층(3) / 인증 헤더, ESP 무결성과 인증을 보장하는 인증 헤더 AH와 기밀성을 보장하는 ESP를 이용, 양 종단(End point) 간에 보안 서비스를 제공하는 터널링 프로토콜
	SSL	Secure Sockets Layer	클라이언트와 서버 간의 웹 데이터 암호화, 기밀성 상호 인증 및 전송 시, 데이터 무결성을 보장하는 보안 프로토콜 (TLS는 SSL 3.0을 기초로 안전성 높임)
	TLS	Transport Layer Security	
	S-HTTP	Secure Hypertext Transfer Protocol	웹상에서 네트워크 트래픽을 암호화 클라이언트와 서버 간에 전송되는 모든 메시지를 암호화
인터페이스 구현 검증도구⑥	xUnit	자바, C++, .Net등 다양한 언어 지원, 단위테스트 프레임워크	
	STAF	Software Testing Automation Framework, 소프트웨어 테스트 자동 프레임워크 서비스 호출, 컴포넌트 재사용 등 다양한 환경 지원하는 테스트 프레임워크	
	FitNesse	웹 기반 테스트 케이스, 설계/실행/결과 확인 등을 지원하는 테스트 프레임워크	
	NTAF	FitNesse(협업가능) + STAF(재사용, 확장성) = 통합 테스트 자동화 프레임워크	
	Selenium	다양한 브라우저 지원 및 개발언어를 지원하는 웹 애플리케이션 테스트 프레임워크	
	watir	루비Ruby 기반 웹 애플리케이션 테스트 프레임워크	
인터페이스 감시도구②	스카우터 SCOUTER: app 모니터링, DB Agent로 오픈소스 DB모니터링 기능, 인터페이스 감시 기능 제니퍼 Jennifer: app 개발부터 테스트, 오픈, 운영, 안정화까지 전 생애주기 단계 모니터링, 분석 API		

Dos공격 Denial of Service	시스템을 공격하여, 해당 시스템의 자원을 고갈시킴 => 의도된 용도대로 사용하지 못하게 하는 공격 가용성을 해침		
DoS 공격종류~⑦	TCP SYN 플러딩	SYN Flooding	서버의 동시 가용 사용자수를 SYN패킷으로 점유, 사용자가 서버를 사용하지 못하게 함
	UDP 플러딩	UDP Flooding	대량의 UDP패킷을 임의의 포트 번호로 전송, 응답 메시지 생성=>지속적 자원 고갈
	ICMP 플러딩 스머프/스머핑	Smurf/Smurfing	출발지 주소를 대상의 IP로 설정, 다량의 Echo패킷 분산 서비스 거부 공격이다.
	죽음의 핑	PoD, Ping of Death	ICMP패킷을 비정상적으로 크게 전송함
	랜드 어택	Land Attack	출발지 IP와 목적지 IP 같은 패킷 주소로 보냄
	티어 드롭	Tear Drop	IP패킷의 재조합 방해, 수신 시스템 문제 일으킴
	봉크/보잉크	Bonk, Boink	프로토콜의 오류 제어를 이용한 공격기법
	DDos공격 Distributed Denial of Service	여러 대의 공격자를 분산 배치하여, 동시에 동작하게 함으로써 특정 사이트를 공격하는 기법	
DDoS 공격도구③	Trinoo	많은 소스로부터 통합, UDP flood 서비스 거부 공격을 유발	
	Tribe Flood Network	많은 소스에서 하나 혹은 여러 개의 목표 시스템에 서비스 공격	
	Stacheldraht	분산 서비스 거부 에이전트 역할을 하는 Linux, Solaris용 멀웨어 도구	
DoS DDoS DRDos	DoS: 직접 공격, 한 사람에 의해 공격 DDoS: 공격하도록 지시, 수많은 감염 호스트를 통해 공격 DRDos: 출발지 IP를 공격Port Scanning대상 IP로 위조, 다량의 응답을 반사시켜 공격		
보안 관련 용어, 앱&네트워크 공격 유형 & 방법	HTTP GET 플러딩	HTTP Get을 지속적으로 요청, 처리 로직의 과부하를 유발	
	Slow HTTP Get/Post	소량의 데이터를 느린 속도로 전송하여 장시간 세션을 유지	
	Slowloris	HTTP Get, 헤더의 최종 끝을 알리는 개행 문자열 전송x, 장시간 연결	
	Hulk DoS	공격대상 URL을 지속적으로 변경, 차단정책을 우회하는 Get Flooding	
	Hash DoS	다량의 파라미터를 POST, 다수의 해시 충돌 발생	
	스니핑 Sniffing	네트워크로 전송되는 패킷을 훑쳐보는 도구	
	세션 하이재킹 hijacking	인증을 받은 세션 연결을 빼앗는 공격	
	IP 스푸핑 Spoofing	자신의 IP를 속이는 행위	
	ARP 스푸핑	클라이언트의 MAC주소를 중간에 공격자의 MAC주소로 변조하여 속임	
	SQL injection	SQL문 삽입해 의도하지 않은 명령을 수행, 허용되지 않은 정보에 접근	
	XXS, Corss Site Script	검증되지 않은 외부 입력 데이터 포함 웹페이지 열람, 스크립트 실행	
	CSRF, Cross-Stie Request Forgery	사용자 의지와 무관, 공격자 의도대로 사이트를 공격하게 함	
	ICMP 리다이렉트 Redirect	ICMP redirect를 위조한 메시지 생성, 라우팅 경로를 변조함	
	버퍼 Buffer 오버플로우	메모리 할당 버퍼 크기를 초과하는 양의 데이터 입력, 악성코드 실행	
	백도어 Backdoor	허가받지 않고 시스템에 접속, 정상적인 인증 절차를 우회함	
	트로이목마	악성 루틴이 숨어있는 프로그램, 실행하면 악성 코드를 실행함	
	큐싱 Qshing	QR코드(+피싱), 악성 앱을 다운 유도, 금융정보 등을 빼내는 피싱	
	스미싱 Smishing	SMS(+피싱), 개인 비밀정보 요구, 소액결제 요구	
	봇넷 Botnet	악성 프로그램이 감염되어 있는 컴퓨터들이 네트워크로 연결된 형태	
	제로데이 공격 Zero day	보안 취약점 발견 후, 공표되기 전에 취약점을 악용하는 기법	
	악성 봇 Malicious Bot	해커의 명령에 의해 원격 제어 또는 실행이 가능한 프로그램 또는 코드	
	랜섬웨어 Ransomware	시스템의 파일을 암호화, 인질로 잡고 몸값을 요구하는 소프트웨어	
	이블 트윈 Evil Twin	핫스팟에 연결한 무선 사용자들의 정보를 탈취하는 공격	
	웜 Worm	스스로 복제해 네트워크 등 연결을 통하여 전파하는 악성 소프트웨어	
	레인보우 테이블	해시테이블, 해시 암호화를 매칭해 복호화, 패스워드 크래킹 / salt	
	워터링홀 watering hole	합법적인 사이트를 미리 악성코드에 감염시켜 희생자를 감염 /제로데이	
	메타스플로잇Metasploit	공격코드, 보안테스팅 등을 제공하는 오픈소스 도구	
	공급망 공격 Supply Chain Attack	개발사의 네트워크에 침투, 악의적 코드 삽입, 업데이트 배포	
	APT, Advanced Persistent Threat	다양한 수단을 통해 지속적이고 지능적으로 맞춤 공격	
	사이버 킬체인 Cyber Kill Chain	공격형 방위 시스템, APT공격방어 분석 모델	
	스피어피싱 Spear-Phishing	발송 메일의 링크, 첨부파일을 클릭하도록 유도, 개인정보 탈취	
	포트 스캐닝 Port Scanning	서버에 열려있는 포트를 확인하고 취약점을 이용한다.	

배치 프로그램 Batch Program	사용자와의 상호 작용 없이, 일련의 작업들을 작업 단위로 묶어, 정기적으로 반복수행하거나, 정해진 규칙에 따라 일괄 처리하는 방법 // 배치 스케줄러: 일괄처리 작업이 설정된 주기에 맞춰 자동으로 수행되도록 지원하는 도구 ex) 스프링 배치, 퀴즈 스케줄러																										
소프트웨어 개발 보안의 요소③	<table><tr><td>기밀성</td><td>Confidentiality</td><td>시스템 내 정보와 자원은, 인가된 사용자만 접근</td></tr><tr><td>무결성</td><td>Integrity</td><td>시스템 내 정보는, 인가된 사용자만 수정</td></tr><tr><td>가용성</td><td>Availability</td><td>인가받은 사용자는 시스템 내 정보와 자원을 언제든지 사용</td></tr></table>	기밀성	Confidentiality	시스템 내 정보와 자원은, 인가된 사용자만 접근	무결성	Integrity	시스템 내 정보는, 인가된 사용자만 수정	가용성	Availability	인가받은 사용자는 시스템 내 정보와 자원을 언제든지 사용																	
기밀성	Confidentiality	시스템 내 정보와 자원은, 인가된 사용자만 접근																									
무결성	Integrity	시스템 내 정보는, 인가된 사용자만 수정																									
가용성	Availability	인가받은 사용자는 시스템 내 정보와 자원을 언제든지 사용																									
서버 인증의 기능?	스니핑 방지(SSL인증서 설치), 피싱 방지, 데이터 변조 방지, 기업 신뢰도 향상(기업 인증)																										
인증 기술 유형④	<table><tr><td>지식기반 인증</td><td>사용자가 기억하고 있는 지식</td><td>ID/PW</td></tr><tr><td>소지기반 인증</td><td>소지하고 있는 사용자 물품</td><td>공인인증서, OTP</td></tr><tr><td>생체기반 인증</td><td>고유한 사용자의 생체 정보</td><td>홍채, 얼굴, 지문</td></tr><tr><td>특정기반 인증</td><td>사용자의 특징을 활용</td><td>서명, 몸짓</td></tr></table>			지식기반 인증	사용자가 기억하고 있는 지식	ID/PW	소지기반 인증	소지하고 있는 사용자 물품	공인인증서, OTP	생체기반 인증	고유한 사용자의 생체 정보	홍채, 얼굴, 지문	특정기반 인증	사용자의 특징을 활용	서명, 몸짓												
지식기반 인증	사용자가 기억하고 있는 지식	ID/PW																									
소지기반 인증	소지하고 있는 사용자 물품	공인인증서, OTP																									
생체기반 인증	고유한 사용자의 생체 정보	홍채, 얼굴, 지문																									
특정기반 인증	사용자의 특징을 활용	서명, 몸짓																									
접근 통제 기법 ④	<table><tr><td>식별</td><td>Identification</td><td>자신이 누구라고 시스템에 밝힘</td><td>/ 나 누구</td></tr><tr><td>인증</td><td>Authentication</td><td>주체의 신원을 검증하기 위한 활동</td><td>/ 신원 검증</td></tr><tr><td>인가</td><td>Authorization</td><td>인증된 주체에 접근을 허용하는 활동</td><td>/ 접근 허용</td></tr><tr><td>책임추적성</td><td>Accountability</td><td>주체의 접근을 추적하고 행동을 기록</td><td>/ 추적 기록</td></tr></table>			식별	Identification	자신이 누구라고 시스템에 밝힘	/ 나 누구	인증	Authentication	주체의 신원을 검증하기 위한 활동	/ 신원 검증	인가	Authorization	인증된 주체에 접근을 허용하는 활동	/ 접근 허용	책임추적성	Accountability	주체의 접근을 추적하고 행동을 기록	/ 추적 기록								
식별	Identification	자신이 누구라고 시스템에 밝힘	/ 나 누구																								
인증	Authentication	주체의 신원을 검증하기 위한 활동	/ 신원 검증																								
인가	Authorization	인증된 주체에 접근을 허용하는 활동	/ 접근 허용																								
책임추적성	Accountability	주체의 접근을 추적하고 행동을 기록	/ 추적 기록																								
접근 통제 유형 ③ Access Control	<table><tr><td>임의적 접근통제</td><td>DAC</td><td>Discretionary Access Control</td><td>개인 또는 그룹 Identity에 근거, 접근 통제 권한자는 권한을 이양할 수 있음(넘겨줌)</td></tr><tr><td>강제적 접근통제</td><td>MAC</td><td>Mandatory Access Control</td><td>주체가 갖는 허가 권한 / 보안 등급 기간 모든 주체, 객체에 대해 일정함</td></tr><tr><td>역할 기반 접근통제</td><td>RBAC</td><td>Role Based Access Control</td><td>DAC와 MAC의 단점 보완, 역할별로 권한 부여 Admin기능 / Admin리뷰 / System level기능</td></tr></table>			임의적 접근통제	DAC	Discretionary Access Control	개인 또는 그룹 Identity에 근거, 접근 통제 권한자는 권한을 이양할 수 있음(넘겨줌)	강제적 접근통제	MAC	Mandatory Access Control	주체가 갖는 허가 권한 / 보안 등급 기간 모든 주체, 객체에 대해 일정함	역할 기반 접근통제	RBAC	Role Based Access Control	DAC와 MAC의 단점 보완, 역할별로 권한 부여 Admin기능 / Admin리뷰 / System level기능												
임의적 접근통제	DAC	Discretionary Access Control	개인 또는 그룹 Identity에 근거, 접근 통제 권한자는 권한을 이양할 수 있음(넘겨줌)																								
강제적 접근통제	MAC	Mandatory Access Control	주체가 갖는 허가 권한 / 보안 등급 기간 모든 주체, 객체에 대해 일정함																								
역할 기반 접근통제	RBAC	Role Based Access Control	DAC와 MAC의 단점 보완, 역할별로 권한 부여 Admin기능 / Admin리뷰 / System level기능																								
접근 통제 보호 모델 ③	<table><tr><td>벨-라파둘라 BLP Bell-LaPadula Confidentiality Model</td><td>기밀성 O</td><td colspan="2">미 국방부, 기밀성O / 군사적 목적 충족, 최초 모델 No read up / No write down</td></tr><tr><td>비바 BIBA</td><td>무결성 O 기밀성 X</td><td colspan="2">BLP단점 보완 무결성 모델 / 기밀성X No read down / No write up</td></tr><tr><td>클락-윌슨 Clark-Wilson</td><td>무결성↑ 기밀성↓</td><td colspan="2">상업 환경 적합, 불법수정 방지 / 기밀성보다 무결성</td></tr></table>			벨-라파둘라 BLP Bell-LaPadula Confidentiality Model	기밀성 O	미 국방부, 기밀성O / 군사적 목적 충족, 최초 모델 No read up / No write down		비바 BIBA	무결성 O 기밀성 X	BLP단점 보완 무결성 모델 / 기밀성X No read down / No write up		클락-윌슨 Clark-Wilson	무결성↑ 기밀성↓	상업 환경 적합, 불법수정 방지 / 기밀성보다 무결성													
벨-라파둘라 BLP Bell-LaPadula Confidentiality Model	기밀성 O	미 국방부, 기밀성O / 군사적 목적 충족, 최초 모델 No read up / No write down																									
비바 BIBA	무결성 O 기밀성 X	BLP단점 보완 무결성 모델 / 기밀성X No read down / No write up																									
클락-윌슨 Clark-Wilson	무결성↑ 기밀성↓	상업 환경 적합, 불법수정 방지 / 기밀성보다 무결성																									
암호 알고리즘? Encryption Algorithm	데이터의 무결성, 기밀성 확보를 위해 정보를 쉽게 해독할 수 없는 형태로 변환하는 기법 (양방향: 대칭키+비대칭키 / 일방향: 해시)																										
블록 암호방식? Block Cipher	고정 길이의 블록을 암호화하는 방식																										
대칭키 암호화 알고리즘 Symmetric-key Algorithm	<table><tr><td>DES</td><td>Data Encryption Standard</td><td colspan="2">데이터 암호화 표준, 1975년 IBM개발, 대칭 키 기반</td></tr><tr><td>AES</td><td>Advanced Encryption Standard</td><td colspan="2">2001년 미국 표준기술 연구소 NIST 개발</td></tr><tr><td>SEED</td><td>시드 블록 암호 알고리즘</td><td colspan="2">1999년 한국인터넷진흥원 KISA 개발</td></tr><tr><td>ARIA</td><td>Academy, Research Institute, Agency</td><td colspan="2">한국 국가보안기술연구소, 국가표준 암호알고리즘</td></tr><tr><td>IDEA</td><td>International Data Encryption Algorithm</td><td colspan="2">DES대체, 국제 암호 알고리즘, 스위스 연방기술기관</td></tr><tr><td>LSFR</td><td>Linear Feedback Shift Register</td><td colspan="2">스트림 암호화 알고리즘, 선형 귀환 시프트 레지스터</td></tr></table>			DES	Data Encryption Standard	데이터 암호화 표준, 1975년 IBM개발, 대칭 키 기반		AES	Advanced Encryption Standard	2001년 미국 표준기술 연구소 NIST 개발		SEED	시드 블록 암호 알고리즘	1999년 한국인터넷진흥원 KISA 개발		ARIA	Academy, Research Institute, Agency	한국 국가보안기술연구소, 국가표준 암호알고리즘		IDEA	International Data Encryption Algorithm	DES대체, 국제 암호 알고리즘, 스위스 연방기술기관		LSFR	Linear Feedback Shift Register	스트림 암호화 알고리즘, 선형 귀환 시프트 레지스터	
DES	Data Encryption Standard	데이터 암호화 표준, 1975년 IBM개발, 대칭 키 기반																									
AES	Advanced Encryption Standard	2001년 미국 표준기술 연구소 NIST 개발																									
SEED	시드 블록 암호 알고리즘	1999년 한국인터넷진흥원 KISA 개발																									
ARIA	Academy, Research Institute, Agency	한국 국가보안기술연구소, 국가표준 암호알고리즘																									
IDEA	International Data Encryption Algorithm	DES대체, 국제 암호 알고리즘, 스위스 연방기술기관																									
LSFR	Linear Feedback Shift Register	스트림 암호화 알고리즘, 선형 귀환 시프트 레지스터																									
비대칭키 암호화 알고리즘 (공개키 알고리즘)	<table><tr><td>디피-헬만</td><td colspan="3">최초의 공개키 알고리즘</td></tr><tr><td>RSA</td><td colspan="3">1977년 MIT 개발</td></tr><tr><td>ElGamal</td><td colspan="3">1984년 개발</td></tr><tr><td>ECC</td><td colspan="3">1985년 RSA대안</td></tr></table>			디피-헬만	최초의 공개키 알고리즘			RSA	1977년 MIT 개발			ElGamal	1984년 개발			ECC	1985년 RSA대안										
디피-헬만	최초의 공개키 알고리즘																										
RSA	1977년 MIT 개발																										
ElGamal	1984년 개발																										
ECC	1985년 RSA대안																										
해시 암호 방식 (일방향 암호 방식)	<table><tr><td>MAC</td><td>Message Authentication Code</td><td>키를 사용하지 않는 메시지 인증 코드</td><td>메시지 무결성+송신자 인증</td></tr><tr><td>MDC</td><td>Modification Detection Code</td><td>키를 사용하지 않는 변경 감지 코드</td><td>메시지 무결성</td></tr></table>			MAC	Message Authentication Code	키를 사용하지 않는 메시지 인증 코드	메시지 무결성+송신자 인증	MDC	Modification Detection Code	키를 사용하지 않는 변경 감지 코드	메시지 무결성																
MAC	Message Authentication Code	키를 사용하지 않는 메시지 인증 코드	메시지 무결성+송신자 인증																								
MDC	Modification Detection Code	키를 사용하지 않는 변경 감지 코드	메시지 무결성																								
해시 암호화 알고리즘	<table><tr><td>MD5</td><td colspan="3">MD4 개선, 파일의 무결성 검사에 사용</td></tr><tr><td>SHA-1</td><td colspan="3">1993년 NSA에 의해 미국 정부 표준 지정</td></tr><tr><td>SHA-256/384/512</td><td colspan="3">256비트의 해시값을 생성하는 해시함수</td></tr><tr><td>HAS-160</td><td colspan="3">국내 표준 서명 알고리즘</td></tr><tr><td>HAVAL</td><td colspan="3">메시지를 1024bit 블록으로 나눔</td></tr></table>			MD5	MD4 개선, 파일의 무결성 검사에 사용			SHA-1	1993년 NSA에 의해 미국 정부 표준 지정			SHA-256/384/512	256비트의 해시값을 생성하는 해시함수			HAS-160	국내 표준 서명 알고리즘			HAVAL	메시지를 1024bit 블록으로 나눔						
MD5	MD4 개선, 파일의 무결성 검사에 사용																										
SHA-1	1993년 NSA에 의해 미국 정부 표준 지정																										
SHA-256/384/512	256비트의 해시값을 생성하는 해시함수																										
HAS-160	국내 표준 서명 알고리즘																										
HAVAL	메시지를 1024bit 블록으로 나눔																										

시큐어 코딩이란? Secure Coding	설계 및 구현 단계에서 해킹 등의 공격을 유발할 수 있는 잠재적 보안 취약점을 사전에 제거																																							
보안 운영체제란? Secure OS	컴퓨터 운영체제의 커널에 보안 기능을 추가한 솔루션																																							
애플리케이션 테스트 원리⑤	<table><tr><td>완벽한 테스트는 불가능</td><td colspan="3">결함을 줄일 수는 있으나, 없다고 증명할 수 없음</td></tr><tr><td>파레토 법칙 Pareto</td><td colspan="3">20% 코드에서 80%의 결함이 발견됨</td></tr><tr><td>살충제 패러독스</td><td colspan="3">동일한 테스트를 반복하면 더 이상 결함이 발견되지 않음</td></tr><tr><td>정확 의존성</td><td colspan="3">소프트웨어 성격에 맞는 테스트 실시</td></tr><tr><td>오류-부재의 궤변</td><td colspan="3">요구사항을 충족시키지 못하면 결함이 없어도 품질이 높다고 볼 수 없다.</td></tr></table>				완벽한 테스트는 불가능	결함을 줄일 수는 있으나, 없다고 증명할 수 없음			파레토 법칙 Pareto	20% 코드에서 80%의 결함이 발견됨			살충제 패러독스	동일한 테스트를 반복하면 더 이상 결함이 발견되지 않음			정확 의존성	소프트웨어 성격에 맞는 테스트 실시			오류-부재의 궤변	요구사항을 충족시키지 못하면 결함이 없어도 품질이 높다고 볼 수 없다.																		
완벽한 테스트는 불가능	결함을 줄일 수는 있으나, 없다고 증명할 수 없음																																							
파레토 법칙 Pareto	20% 코드에서 80%의 결함이 발견됨																																							
살충제 패러독스	동일한 테스트를 반복하면 더 이상 결함이 발견되지 않음																																							
정확 의존성	소프트웨어 성격에 맞는 테스트 실시																																							
오류-부재의 궤변	요구사항을 충족시키지 못하면 결함이 없어도 품질이 높다고 볼 수 없다.																																							
프로그램 실행 여부에 따른 분류	<table><tr><td>정적 테스트</td><td>Static</td><td>테스트 대상을 실행X, 구조 분석/논리성 검증</td><td>리뷰, 정적분석</td></tr><tr><td>동적 테스트</td><td>Dynamic</td><td>소프트웨어를 실행, 결함을 검출함</td><td>화이트&블랙박스/경험기반</td></tr></table>				정적 테스트	Static	테스트 대상을 실행X, 구조 분석/논리성 검증	리뷰, 정적분석	동적 테스트	Dynamic	소프트웨어를 실행, 결함을 검출함	화이트&블랙박스/경험기반																												
정적 테스트	Static	테스트 대상을 실행X, 구조 분석/논리성 검증	리뷰, 정적분석																																					
동적 테스트	Dynamic	소프트웨어를 실행, 결함을 검출함	화이트&블랙박스/경험기반																																					
화이트박스 테스트 White-Box Test (구조,결정,조건,흐름)	<table><tr><td>구문 커버리지</td><td colspan="3">모든 구문 / 프로그램 내 모든 명령문을 적어도 한 번 수행</td></tr><tr><td>결정 커버리지</td><td colspan="3">모든 분기 / 결정 포인트 내 전체 조건식이 적어도 한번은 참과 거짓 수행 커버리지</td></tr><tr><td>조건 커버리지</td><td colspan="3">모든 조건 / 결정 포인트 내 각 개별 조건식이 적어도 한번은 참과 거짓, 커버리지</td></tr><tr><td>조건/결정 커버리지</td><td colspan="3">전체 조건식&개별 조건식 모두 참, 거짓</td></tr><tr><td>변경 조건/결정 커버리지</td><td colspan="3">개별 조건식이 다른 개별 조건식에 영향X, 전체 조건식에 독립적 영향</td></tr><tr><td>다중 조건 커버리지</td><td colspan="3">모든 개별 조건식 / 모든 가능한 조합을 100% 보장</td></tr><tr><td>기본 경로 커버리지</td><td colspan="3">수행가능한 모든 경로를 테스트</td></tr><tr><td>제어 흐름 테스트</td><td colspan="3">제어 구조를 그래프 형태로, 내부 로직 테스트</td></tr><tr><td>데이터 흐름 테스트</td><td colspan="3">제어 흐름 그래프에 사용현황을 추가한 테스트 기법</td></tr></table>				구문 커버리지	모든 구문 / 프로그램 내 모든 명령문을 적어도 한 번 수행			결정 커버리지	모든 분기 / 결정 포인트 내 전체 조건식이 적어도 한번은 참과 거짓 수행 커버리지			조건 커버리지	모든 조건 / 결정 포인트 내 각 개별 조건식이 적어도 한번은 참과 거짓, 커버리지			조건/결정 커버리지	전체 조건식&개별 조건식 모두 참, 거짓			변경 조건/결정 커버리지	개별 조건식이 다른 개별 조건식에 영향X, 전체 조건식에 독립적 영향			다중 조건 커버리지	모든 개별 조건식 / 모든 가능한 조합을 100% 보장			기본 경로 커버리지	수행가능한 모든 경로를 테스트			제어 흐름 테스트	제어 구조를 그래프 형태로, 내부 로직 테스트			데이터 흐름 테스트	제어 흐름 그래프에 사용현황을 추가한 테스트 기법		
구문 커버리지	모든 구문 / 프로그램 내 모든 명령문을 적어도 한 번 수행																																							
결정 커버리지	모든 분기 / 결정 포인트 내 전체 조건식이 적어도 한번은 참과 거짓 수행 커버리지																																							
조건 커버리지	모든 조건 / 결정 포인트 내 각 개별 조건식이 적어도 한번은 참과 거짓, 커버리지																																							
조건/결정 커버리지	전체 조건식&개별 조건식 모두 참, 거짓																																							
변경 조건/결정 커버리지	개별 조건식이 다른 개별 조건식에 영향X, 전체 조건식에 독립적 영향																																							
다중 조건 커버리지	모든 개별 조건식 / 모든 가능한 조합을 100% 보장																																							
기본 경로 커버리지	수행가능한 모든 경로를 테스트																																							
제어 흐름 테스트	제어 구조를 그래프 형태로, 내부 로직 테스트																																							
데이터 흐름 테스트	제어 흐름 그래프에 사용현황을 추가한 테스트 기법																																							
블랙박스 테스트 Black-Box Test⑤ (동원인비오경)	<table><tr><td>동등 분할 테스트 Equivalence Partitioning</td><td colspan="3">입력 자료에 초점, 도메인별 유효값/무효값 그룹핑</td></tr><tr><td>원인-결과 그래프 테스트 Cause Effect Graph</td><td colspan="3">그래프, 체계적 분석 후 효용성 높은 케이스</td></tr><tr><td>비교 테스트 Comparison Testing</td><td colspan="3">여러 버전에 동일한 테스트 자료, 동일한 결과 테스트</td></tr><tr><td>오류 예측 기법 Error Guessing</td><td colspan="3">데이터 확인 검사</td></tr><tr><td>경계값 분석 테스트 Boundary Value Analysis</td><td colspan="3">동치분할기법 보완, 입력 조건의 경계값</td></tr><tr><td colspan="4">결정 테이블, 상태전이, 유스케이스, 분류 트리, 페어와이즈pairwises(조합)</td></tr></table>				동등 분할 테스트 Equivalence Partitioning	입력 자료에 초점, 도메인별 유효값/무효값 그룹핑			원인-결과 그래프 테스트 Cause Effect Graph	그래프, 체계적 분석 후 효용성 높은 케이스			비교 테스트 Comparison Testing	여러 버전에 동일한 테스트 자료, 동일한 결과 테스트			오류 예측 기법 Error Guessing	데이터 확인 검사			경계값 분석 테스트 Boundary Value Analysis	동치분할기법 보완, 입력 조건의 경계값			결정 테이블, 상태전이, 유스케이스, 분류 트리, 페어와이즈pairwises(조합)															
동등 분할 테스트 Equivalence Partitioning	입력 자료에 초점, 도메인별 유효값/무효값 그룹핑																																							
원인-결과 그래프 테스트 Cause Effect Graph	그래프, 체계적 분석 후 효용성 높은 케이스																																							
비교 테스트 Comparison Testing	여러 버전에 동일한 테스트 자료, 동일한 결과 테스트																																							
오류 예측 기법 Error Guessing	데이터 확인 검사																																							
경계값 분석 테스트 Boundary Value Analysis	동치분할기법 보완, 입력 조건의 경계값																																							
결정 테이블, 상태전이, 유스케이스, 분류 트리, 페어와이즈pairwises(조합)																																								
테스트 시각에 따른 분류	<table><tr><td>검증</td><td>Verification</td><td>소프트웨어 개발 과정을 테스트</td><td>개발자+시험자 시각</td></tr><tr><td>확인</td><td>Validation</td><td>소프트웨어 결과를 테스트</td><td>사용자 시각</td></tr></table>				검증	Verification	소프트웨어 개발 과정을 테스트	개발자+시험자 시각	확인	Validation	소프트웨어 결과를 테스트	사용자 시각																												
검증	Verification	소프트웨어 개발 과정을 테스트	개발자+시험자 시각																																					
확인	Validation	소프트웨어 결과를 테스트	사용자 시각																																					
테스트 목적에 따른 분류⑥ (회회구안병성)	<table><tr><td>회복 테스트</td><td>Recovery</td><td colspan="2">고의 실패, 시스템의 정상적 복구 여부를 테스트하는 기법</td></tr><tr><td>회귀 테스트</td><td>Regression</td><td colspan="2">변경 또는 수정된 코드에 새로운 결함이 없음을 확인하는 테스트</td></tr><tr><td>구조 테스트</td><td>Structure</td><td colspan="2">내부 논리 경로, 소스코드의 복잡도를 평가하는 테스트 기법</td></tr><tr><td>안전 테스트</td><td>Security</td><td colspan="2">불법 소프트웨어가 시스템을 파괴하지 못하도록, 보안결함을 미리 점검</td></tr><tr><td>병행 테스트</td><td>Parallel</td><td colspan="2">변경 시스템과 기존 시스템에 동일 데이터를 입력 후 결과를 비교</td></tr><tr><td>성능 테스트</td><td>Performance</td><td colspan="2">시스템 응답시간, 처리량, 반응 속도 등을 테스트하는 기법</td></tr></table>				회복 테스트	Recovery	고의 실패, 시스템의 정상적 복구 여부를 테스트하는 기법		회귀 테스트	Regression	변경 또는 수정된 코드에 새로운 결함이 없음을 확인하는 테스트		구조 테스트	Structure	내부 논리 경로, 소스코드의 복잡도를 평가하는 테스트 기법		안전 테스트	Security	불법 소프트웨어가 시스템을 파괴하지 못하도록, 보안결함을 미리 점검		병행 테스트	Parallel	변경 시스템과 기존 시스템에 동일 데이터를 입력 후 결과를 비교		성능 테스트	Performance	시스템 응답시간, 처리량, 반응 속도 등을 테스트하는 기법													
회복 테스트	Recovery	고의 실패, 시스템의 정상적 복구 여부를 테스트하는 기법																																						
회귀 테스트	Regression	변경 또는 수정된 코드에 새로운 결함이 없음을 확인하는 테스트																																						
구조 테스트	Structure	내부 논리 경로, 소스코드의 복잡도를 평가하는 테스트 기법																																						
안전 테스트	Security	불법 소프트웨어가 시스템을 파괴하지 못하도록, 보안결함을 미리 점검																																						
병행 테스트	Parallel	변경 시스템과 기존 시스템에 동일 데이터를 입력 후 결과를 비교																																						
성능 테스트	Performance	시스템 응답시간, 처리량, 반응 속도 등을 테스트하는 기법																																						
성능테스트 Performance Testing 상세 유형④	<table><tr><td>부하 테스트</td><td>Load Testing</td><td colspan="2">시스템에 부가를 증가시켜 임계점을 찾는 테스트</td></tr><tr><td>강도 테스트</td><td>Stress Testing</td><td colspan="2">임계점 이상의 부하를 가해, 비정상적 상황의 처리 테스트</td></tr><tr><td>내구성 테스트</td><td>Endurance Testing</td><td colspan="2">오랜 시간 높은 부하를 가해, 시스템의 반응 테스트</td></tr><tr><td>스파이크 테스트</td><td>Spike Testing</td><td colspan="2">짧은 시간에 사용자가 몰릴 때, 시스템의 반응 측정</td></tr></table>				부하 테스트	Load Testing	시스템에 부가를 증가시켜 임계점을 찾는 테스트		강도 테스트	Stress Testing	임계점 이상의 부하를 가해, 비정상적 상황의 처리 테스트		내구성 테스트	Endurance Testing	오랜 시간 높은 부하를 가해, 시스템의 반응 테스트		스파이크 테스트	Spike Testing	짧은 시간에 사용자가 몰릴 때, 시스템의 반응 측정																					
부하 테스트	Load Testing	시스템에 부가를 증가시켜 임계점을 찾는 테스트																																						
강도 테스트	Stress Testing	임계점 이상의 부하를 가해, 비정상적 상황의 처리 테스트																																						
내구성 테스트	Endurance Testing	오랜 시간 높은 부하를 가해, 시스템의 반응 테스트																																						
스파이크 테스트	Spike Testing	짧은 시간에 사용자가 몰릴 때, 시스템의 반응 측정																																						

테스트 종류에 따른 분류③ (명구경)	<table><tr><td>명세 기반 테스트</td><td>블랙박스</td><td>요구사항 명세서</td></tr><tr><td>구현 기반 테스트</td><td>화이트박스</td><td>내부 논리 흐름</td></tr><tr><td>경험 기반 테스트</td><td>블랙박스</td><td>테스터의 경험을 기반</td></tr></table>	명세 기반 테스트	블랙박스	요구사항 명세서	구현 기반 테스트	화이트박스	내부 논리 흐름	경험 기반 테스트	블랙박스	테스터의 경험을 기반							
명세 기반 테스트	블랙박스	요구사항 명세서															
구현 기반 테스트	화이트박스	내부 논리 흐름															
경험 기반 테스트	블랙박스	테스터의 경험을 기반															
테스트 케이스란? <u>Test Case</u>	사용자의 <u>요구사항</u> 을 정확하게 준수했는지 <u>확인</u> 하기 위해 <u>설계</u> 된, <u>테스트 항목</u> 에 대한 명세서																
테스트 오라클이란? <u>Test Oracle</u>	테스트 결과가 올바른지 판단하기 위해 사전에 정의된 참 값을 대입하여 비교하는 방법																
테스트 오라클 종류④ (참샘휴일)	<table><tr><td>참 오라클</td><td>True Oracle</td><td>모든 입력값에 대해, 기대하는 결과를 제공</td></tr><tr><td>샘플링 오라클</td><td>Sampling Oracle</td><td>특정 몇 개의 입력값에 대해, 기대하는 결과를 제공</td></tr><tr><td>휴리스틱 오라클</td><td>Heuristic Oracle</td><td>특정 입력값에 대해 올바른 결과 제공, 나머지 추정</td></tr><tr><td>일관성 검사</td><td>Consistent Oracle</td><td>애플리케이션 변경이 있을 때, 결과값 동일 여부 확인</td></tr></table>	참 오라클	True Oracle	모든 입력값에 대해, 기대하는 결과를 제공	샘플링 오라클	Sampling Oracle	특정 몇 개의 입력값에 대해, 기대하는 결과를 제공	휴리스틱 오라클	Heuristic Oracle	특정 입력값에 대해 올바른 결과 제공, 나머지 추정	일관성 검사	Consistent Oracle	애플리케이션 변경이 있을 때, 결과값 동일 여부 확인				
참 오라클	True Oracle	모든 입력값에 대해, 기대하는 결과를 제공															
샘플링 오라클	Sampling Oracle	특정 몇 개의 입력값에 대해, 기대하는 결과를 제공															
휴리스틱 오라클	Heuristic Oracle	특정 입력값에 대해 올바른 결과 제공, 나머지 추정															
일관성 검사	Consistent Oracle	애플리케이션 변경이 있을 때, 결과값 동일 여부 확인															
SDLC <u>V모델</u> 의 순서 ⑧	요분설구 단통시인 요구사항 -> 분석 -> 설계 -> 구현 -> 단위테스트 -> 통합테스트 -> 시스템 테스트 -> 인수테스트 단위(정적/동적), 통합(상향/하향), 시스템(기능/비기능), 인수(알파/베타)																
테스트 레벨이란? <u>Test Level</u>	함께 편성되고 관리되는 테스트 활동 그룹 ex) 단통시인																
테스트 레벨 종류④ (단통시인)	<table><tr><td>단위 테스트</td><td>Unit Testing</td><td>사용자 요구사항에 대한 <u>모듈</u>, 서브루틴</td><td>정적/동적</td></tr><tr><td>통합 테스트</td><td>Integration Testing</td><td>단위테스트가 완료된 <u>모듈 결합</u>하는 과정</td><td>상향/하향</td></tr><tr><td>시스템 테스트</td><td>System Testing</td><td>정상적으로 수행되는지 검증하는 테스트</td><td>기능/비기능</td></tr><tr><td>인수 테스트</td><td>Acceptance Testing</td><td>계약상 <u>요구사항</u>이 만족되는지 확인</td><td>알파/베타</td></tr></table>	단위 테스트	Unit Testing	사용자 요구사항에 대한 <u>모듈</u> , 서브루틴	정적/동적	통합 테스트	Integration Testing	단위테스트가 완료된 <u>모듈 결합</u> 하는 과정	상향/하향	시스템 테스트	System Testing	정상적으로 수행되는지 검증하는 테스트	기능/비기능	인수 테스트	Acceptance Testing	계약상 <u>요구사항</u> 이 만족되는지 확인	알파/베타
단위 테스트	Unit Testing	사용자 요구사항에 대한 <u>모듈</u> , 서브루틴	정적/동적														
통합 테스트	Integration Testing	단위테스트가 완료된 <u>모듈 결합</u> 하는 과정	상향/하향														
시스템 테스트	System Testing	정상적으로 수행되는지 검증하는 테스트	기능/비기능														
인수 테스트	Acceptance Testing	계약상 <u>요구사항</u> 이 만족되는지 확인	알파/베타														
인수 테스트 종류 Acceptance Testing	<table><tr><td><u>알파 테스트</u></td><td>개발자의 장소에서, 개발자와 사용자가 함께 행하는 테스트 기법</td></tr><tr><td><u>베타 테스트</u></td><td>실제 사용자가 대상 소프트웨어를 직접 사용, 피드백을 받는다.</td></tr><tr><td>사용자 인수 테스트</td><td>사용자가 시스템 사용의 적절성 여부를 확인</td></tr><tr><td>운영상의 인수 테스트</td><td>시스템 관리자가 시스템 인수 시 수행하는 테스트 기법</td></tr><tr><td>계약 인수 테스트</td><td>계약 상의 인수/검수 조건 준수여부</td></tr><tr><td>규정 인수 테스트</td><td>소프트웨어가 정부 지침, 법류, 규정에 맞게 개발되었는지 확인</td></tr></table>	<u>알파 테스트</u>	개발자의 장소에서, 개발자와 사용자가 함께 행하는 테스트 기법	<u>베타 테스트</u>	실제 사용자가 대상 소프트웨어를 직접 사용, 피드백을 받는다.	사용자 인수 테스트	사용자가 시스템 사용의 적절성 여부를 확인	운영상의 인수 테스트	시스템 관리자가 시스템 인수 시 수행하는 테스트 기법	계약 인수 테스트	계약 상의 인수/검수 조건 준수여부	규정 인수 테스트	소프트웨어가 정부 지침, 법류, 규정에 맞게 개발되었는지 확인				
<u>알파 테스트</u>	개발자의 장소에서, 개발자와 사용자가 함께 행하는 테스트 기법																
<u>베타 테스트</u>	실제 사용자가 대상 소프트웨어를 직접 사용, 피드백을 받는다.																
사용자 인수 테스트	사용자가 시스템 사용의 적절성 여부를 확인																
운영상의 인수 테스트	시스템 관리자가 시스템 인수 시 수행하는 테스트 기법																
계약 인수 테스트	계약 상의 인수/검수 조건 준수여부																
규정 인수 테스트	소프트웨어가 정부 지침, 법류, 규정에 맞게 개발되었는지 확인																
테스트 시나리오란? <u>Test Scenario</u>	테스트 수행을 위한 여러 테스트 케이스의 집합 테스트 케이스의 동작 순서를 기술한 문서이며, 절차를 명세한 문서																
Mock 객체 생성 프레임워크	객체 지향 프로그램에서는 컴포넌트 테스트 수행 시, 테스트되는 메소드는 다른 클래스 객체에 의존 독립적 컴포넌트 테스트를 위해서 Mock 객체 필요 (aka 더미 객체, 테스트 스텝, 드라이버, 가짜 객체, 테스트 스파이)																
통합 테스트 Integration Test	<table><tr><td>빅뱅 테스트</td><td>모든 모듈을 <u>동시</u>에 통합 후 테스트 수행</td><td>실제 모듈로 테스트</td></tr><tr><td>상향식 테스트</td><td>최하위 모듈부터 점진적으로 상위 모듈과 테스트</td><td>테스트 드라이버 필요</td></tr><tr><td>하향식 테스트</td><td>최상위 모듈부터 점진적으로 하위 모듈과 테스트</td><td>테스트 스텝 필요</td></tr><tr><td>샌드위치 테스트</td><td>상위는 하향식 + 하위는 상향식 테스트</td><td>테스트 스텝+드라이버</td></tr></table>	빅뱅 테스트	모든 모듈을 <u>동시</u> 에 통합 후 테스트 수행	실제 모듈로 테스트	상향식 테스트	최하위 모듈부터 점진적으로 상위 모듈과 테스트	테스트 드라이버 필요	하향식 테스트	최상위 모듈부터 점진적으로 하위 모듈과 테스트	테스트 스텝 필요	샌드위치 테스트	상위는 하향식 + 하위는 상향식 테스트	테스트 스텝+드라이버				
빅뱅 테스트	모든 모듈을 <u>동시</u> 에 통합 후 테스트 수행	실제 모듈로 테스트															
상향식 테스트	최하위 모듈부터 점진적으로 상위 모듈과 테스트	테스트 드라이버 필요															
하향식 테스트	최상위 모듈부터 점진적으로 하위 모듈과 테스트	테스트 스텝 필요															
샌드위치 테스트	상위는 하향식 + 하위는 상향식 테스트	테스트 스텝+드라이버															
테스트 자동화 도구 Test Automation Tool	<div>반복적 테스트 작업을 스크립트 형태로 구현 / 테스트 시간 단축+인력 투입 비용 최소화, 효율적!</div> <table><tr><td>정적 분석 도구</td><td>Static Analysis Tools</td><td>만들어진 애플리케이션을 실행하지 않고 분석</td></tr><tr><td>테스트 실행 도구</td><td>Test Execution Tools</td><td>작성된 테스트 스크립트를 실행</td></tr><tr><td>성능 테스트 도구</td><td>Performance Test Tools</td><td>가상의 사용자 생성, 테스트를 수행</td></tr><tr><td>테스트 통제 도구</td><td>Test Control Tools</td><td>테스트 계획, 관리, 수행, 결함 관리 등 수행</td></tr><tr><td>테스트 하네스 도구</td><td>Test Harness Tools</td><td>실행 환경 시뮬레이션, 모듈&컴포넌트 테스트</td></tr></table>	정적 분석 도구	Static Analysis Tools	만들어진 애플리케이션을 실행하지 않고 분석	테스트 실행 도구	Test Execution Tools	작성된 테스트 스크립트를 실행	성능 테스트 도구	Performance Test Tools	가상의 사용자 생성, 테스트를 수행	테스트 통제 도구	Test Control Tools	테스트 계획, 관리, 수행, 결함 관리 등 수행	테스트 하네스 도구	Test Harness Tools	실행 환경 시뮬레이션, 모듈&컴포넌트 테스트	
정적 분석 도구	Static Analysis Tools	만들어진 애플리케이션을 실행하지 않고 분석															
테스트 실행 도구	Test Execution Tools	작성된 테스트 스크립트를 실행															
성능 테스트 도구	Performance Test Tools	가상의 사용자 생성, 테스트를 수행															
테스트 통제 도구	Test Control Tools	테스트 계획, 관리, 수행, 결함 관리 등 수행															
테스트 하네스 도구	Test Harness Tools	실행 환경 시뮬레이션, 모듈&컴포넌트 테스트															
테스트 하네스 구성요소 Test Harness	테스트 드라이버, 테스트 스텝, 테스트 슈트, 테스트 케이스, 테스트 시나리오, 테스트 스크립트, 목 오브젝트																

소프트웨어 결함 Software Defect	개발자 오류로 인해, 문서 또는 코딩 결점으로 개발자가 설계한 것과 다르게 동작하거나 다른 결과									
테스트 결함 관리	단계별 테스트 수행 후 발생한 결함의 재발 방지 유사 결함 발견 시 처리시간 단축을 위해 결함을 추적하고 관리									
결함 분석 방법③	구체화: 결함의 원인을 찾기 위해, 결함을 발생시킨 입력값, 테스트 절차, 테스트 환경을 명확히 파악 고립화: 입력값, 테스트 절차, 테스트 환경 중 어떤 요소가 결함 발생에 영향을 끼치는지 분석 일반화: 결함 발생에 영향을 주는 요소를 최대한 일반화									
결함 심각도 결함 우선순위	단수 결함(미관)-경미한 결함(표준위반)-보통 결함(사소한 오작동)-주요결함(기능장애)-치명적 결함(데이터 손실) 낮은(Low) -> 보통(Medium) -> 높음(High) -> 결정적(Critical)									
테스트 커버리지③ Test Coverage	테스트 품질 측정 기준 / 주어진 테스트 케이스에 의해 수행되는 소프트웨어 테스트 범위를 측정									
	기능 기반 커버리지: 전체 기능 중, 실제 테스트가 수행된 <u>기능의 수</u> 측정 라인 커버리지: 전체 소스 코드의 라인 수 중, 수행한 <u>소스 코드의 라인 수</u> 측정 코드 커버리지: 소스 코드의 구문, 조건, 결정 등 <u>구조 코드 자체가 얼마나 테스트 되었는지</u> 측정									
애플리케이션 성능 측정 지표 ④	처리량 Throughput: 일정 시간 내에 애플리케이션이 처리하는 일의 양 응답 시간 Response Time: 애플리케이션에 요청을 전달한 시간부터 응답이 도착할 때까지 걸린 시간 경과 시간 Turn Around Time: 애플리케이션에 작업을 의뢰한 시간~처리가 완료될 때까지 걸린 시간 자원 사용률 Resource Usage: 애플리케이션이 의뢰한 작업을 처리하는 동안 (CPU, 메모리, 네트워크) 사용량									
배드 코드? Bad Code	프로그램 로직이 복잡하고 다른 개발자들이 이해하기 어려운 코드 ex) 외계인(에얼리언) 코드, 스파게티 코드, 알 수 없는 변수명, 로직 중복									
클린 코드? Clean Code	가독성이 높고, 단순하며, 의존성을 줄이고, 중복을 최소화하여 깔끔하게 잘 정리된 코드 이해가 쉽고, 개선이 쉽다. 가독성, 단순성, 의존성 최소, 중복성 제거, 추상화									
리팩토링이란? Refactoring	기능을 변경하지 않고, 복잡한 소스코드를 수정, 보완하여 가용성 및 가독성을 높이는 기법									
운영체제 Operating System	사용자가 컴퓨터의 하드웨어를 쉽게 사용할 수 있도록 인터페이스를 제공한다. 제어 프로그램: 감시 Supervisor, 작업관리Job Management, 데이터관리Data Management 제어 장치: 프로그램 카운터, 명령 레지스터, 번지 해독기, 부호기, 번지 레지스터, 기억 레지스터 처리 프로그램: 언어번역, 서비스									
운영체제 성능 평가 요소④	처리능력, 응답시간, 신뢰도, 사용가능도									
메모리 관리 기법 (반입/배치/할당/교체)	반입 기법(When), 배치 기법(Where), 할당 기법(How), 교체 기법(Who)									
메모리 <u>배치</u> 기법	<table><tr><td>최초 적합</td><td>First-Fit</td><td>가용 공간 중 <u>첫 번째 분할</u>에 할당</td></tr><tr><td>최적 적합</td><td>Best-Fit</td><td>가용 공간 중 가장 크기가 <u>비슷한 공간</u>에 할당</td></tr><tr><td>최악 적합</td><td>Worst-Fit</td><td>가용 공간 중 <u>가장 큰 공간</u>에 할당</td></tr></table>	최초 적합	First-Fit	가용 공간 중 <u>첫 번째 분할</u> 에 할당	최적 적합	Best-Fit	가용 공간 중 가장 크기가 <u>비슷한 공간</u> 에 할당	최악 적합	Worst-Fit	가용 공간 중 <u>가장 큰 공간</u> 에 할당
최초 적합	First-Fit	가용 공간 중 <u>첫 번째 분할</u> 에 할당								
최적 적합	Best-Fit	가용 공간 중 가장 크기가 <u>비슷한 공간</u> 에 할당								
최악 적합	Worst-Fit	가용 공간 중 <u>가장 큰 공간</u> 에 할당								

데이터 모델이란? Data Model		소프트웨어 개발과 유지, 보수의 기준 / 추상화된 개념적 모형 현실 세계의 정보들을 컴퓨터로 표현하기 위해 <u>단순화</u> , <u>추상화</u> 하여 <u>체계적</u> 으로 표현																										
개념적 데이터 모델 Conceptual Modeling		현실세계에 대한 인식을 추상적 개념으로 표현한다. 객체 기반 데이터모델이다. 대표적으로 개체 타입과 개체 타입들 간 관계를 이용해 현실 세계를 표현한 E-R모델이 있다.																										
개 념 적 데 이 터 모 델	객체-관계 모델 정의 E-R모델 Entity-Relationship	개체 타입과, 개체 타입들 간의 관계를 이용하여 현실세계를 표현한다. E-R다이어그램을 정의, 개체, 관계, 속성																										
	E-R 다이어그램 시각적 기호	<table border="1"> <thead> <tr> <th>기호</th><th colspan="2">의미</th></tr> </thead> <tbody> <tr> <td>사각형</td><td>개체</td><td>Entity</td></tr> <tr> <td>마름모</td><td>관계</td><td>Relationship</td></tr> <tr> <td>타원</td><td>속성</td><td>Attribute</td></tr> <tr> <td>직사각형</td><td colspan="2">개체 타입</td></tr> <tr> <td>밀줄 타원</td><td colspan="2">기본키 속성</td></tr> <tr> <td>복수 타원</td><td colspan="2">복합 속성</td></tr> <tr> <td><u>n</u> <u>m</u></td><td colspan="2">관계 대응수</td></tr> <tr> <td>선 링크</td><td colspan="2">개체와 속성 연결</td></tr> </tbody> </table>	기호	의미		사각형	개체	Entity	마름모	관계	Relationship	타원	속성	Attribute	직사각형	개체 타입		밀줄 타원	기본키 속성		복수 타원	복합 속성		<u>n</u> <u>m</u>	관계 대응수		선 링크	개체와 속성 연결
기호	의미																											
사각형	개체	Entity																										
마름모	관계	Relationship																										
타원	속성	Attribute																										
직사각형	개체 타입																											
밀줄 타원	기본키 속성																											
복수 타원	복합 속성																											
<u>n</u> <u>m</u>	관계 대응수																											
선 링크	개체와 속성 연결																											
논리적 데이터 모델		개념적 구조를 컴퓨터가 이해하고 처리할 수 있게 변환 데이터 타입과, 데이터 타입들 간 관계를 이용 관계 모델, 계층 모델, 네트워크 모델																										
논 리 데 이 터 모 델	계층 데이터 모델 Hierarchical Data Model	트리형태, 사이클X, 일대다 관계(부모:자식), 다대다 관계 표현X, 개체 삭제 시 연쇄 삭제																										
	관계 모델 Relational Data Model	2차원 테이블, 행과 열, 검색 시 인덱스 사용																										
	네트워크 데이터 모델 Network Data Model	그래프 형태, 사이클O, 일대다 관계(오너:멤버), 다대다 관계 표현, 물리적 구조 알아야																										
물리적 데이터 모델		논리적 데이터 모델에서 레코드의 상세 스펙 등을 기술한 모델 실제 컴퓨터가 데이터에 저장되는 방식을 정의 코드 작성																										
속성의 종류		단순속성, 복합속성, 단일 값 속성, 다중 값 속성, 저장된 속성, 유도된 속성, 키 속성																										
관계 대수 Relational Algebra		일반 집합 연산 : 합집합 / 교집합 / 차집합 / 카티션 프로덕트 Cartesian Product 순수 관계 연산 : 선택 / 프로덕트 / 조인 / 디비전																										
개체 타입		동일한 속성을 가진 개체들의 틀																										

관계형 데이터베이스의 구성 ③		SOC 구조 structure / 연산 operation / 제약조건 constraint:																														
RDBMS 구성요소 SOC를 설명하시오.		구조 Structure: 논리적으로 표현된 <u>개체 타입들 간의 관계</u> , 데이터 구조 (2차원 테이블) 연산 Operation: db에 저장된 데이터를 처리하는 작업에 대한 명세로 db를 조작하는 기본도구 (select, project, join) 제약조건 Constraint: db에 저장될 수 있는 실제 데이터의 논리적인 제약 조건 (개체 무결성, 참조 무결성)																														
릴레이션 개념, 용어 구분		<table border="1"> <tr> <td rowspan="3">표 2d</td><td>릴레이션</td><td>Relation</td><td rowspan="2">데이터의 집합 / 2차원 형식의 표</td></tr> <tr> <td>테이블</td><td>Table</td></tr> <tr> <td>릴레이션 인스턴스</td><td>Relation Instance</td><td>릴레이션 스키마에 저장된 데이터의 집합, (스키마+데이터) 튜플/카디널리티로 구성</td></tr> <tr> <td rowspan="2">행 row</td><td>튜플</td><td>Tuple</td><td>테이블의 행 / 개체 인스턴스 릴레이션에 포함된 튜플들은 모두 다르다.</td></tr> <tr> <td>카디널리티</td><td>Cardinality</td><td>튜플의 개수 / 수시로 변한다. 릴레이션 인스턴스: 실제 저장한 데이터 집합</td></tr> <tr> <td rowspan="4">열 col</td><td>속성</td><td>Attribute</td><td>테이블의 열 / 현실 객체들이 가질 수 있는 값 / 논리적으로 더 이상 쪼갤 수 없는 원자값을 저장</td></tr> <tr> <td>차수</td><td>Degree</td><td>속성의 개수</td></tr> <tr> <td>스키마</td><td>Schema</td><td>릴레이션에 어떤 정보가 담길지 정의 속성/도메인/차수</td></tr> <tr> <td>도메인</td><td>domain</td><td>열 / 속성이 가질 수 있는 모든 가능한 값들의 집합</td></tr> </table>		표 2d	릴레이션	Relation	데이터의 집합 / 2차원 형식의 표	테이블	Table	릴레이션 인스턴스	Relation Instance	릴레이션 스키마에 저장된 데이터의 집합, (스키마+데이터) 튜플/카디널리티로 구성	행 row	튜플	Tuple	테이블의 행 / 개체 인스턴스 릴레이션에 포함된 튜플들은 모두 다르다.	카디널리티	Cardinality	튜플의 개수 / 수시로 변한다. 릴레이션 인스턴스: 실제 저장한 데이터 집합	열 col	속성	Attribute	테이블의 열 / 현실 객체들이 가질 수 있는 값 / 논리적으로 더 이상 쪼갤 수 없는 원자값을 저장	차수	Degree	속성의 개수	스키마	Schema	릴레이션에 어떤 정보가 담길지 정의 속성/도메인/차수	도메인	domain	열 / 속성이 가질 수 있는 모든 가능한 값들의 집합
표 2d	릴레이션	Relation	데이터의 집합 / 2차원 형식의 표																													
	테이블	Table																														
	릴레이션 인스턴스	Relation Instance	릴레이션 스키마에 저장된 데이터의 집합, (스키마+데이터) 튜플/카디널리티로 구성																													
행 row	튜플	Tuple	테이블의 행 / 개체 인스턴스 릴레이션에 포함된 튜플들은 모두 다르다.																													
	카디널리티	Cardinality	튜플의 개수 / 수시로 변한다. 릴레이션 인스턴스: 실제 저장한 데이터 집합																													
열 col	속성	Attribute	테이블의 열 / 현실 객체들이 가질 수 있는 값 / 논리적으로 더 이상 쪼갤 수 없는 원자값을 저장																													
	차수	Degree	속성의 개수																													
	스키마	Schema	릴레이션에 어떤 정보가 담길지 정의 속성/도메인/차수																													
	도메인	domain	열 / 속성이 가질 수 있는 모든 가능한 값들의 집합																													
정규화 목적은? Normalization		관계형 데이터베이스 설계에서 중복성을 최소화해, 이상현상을 방지하고, 일관성과 정확성을 유지한다. 종속 관계를 분석하여 여러개의 릴레이션으로 분해한다.																														
비정규화란? De-Normalization		완벽한 수준의 정규화를 진행하면 일관성, 안정성이 증가하지만 성능이 느려질 수 있기 때문에 성능향상을 위해 <u>릴레이션을 통합, 추가, 분할하는 과정</u> 이다.																														
트랜잭션이란?		하나의 논리적 기능을 정상적으로 수행하기 위한 작업의 기본 단위 활동, 실패, 철회, 부분 완료, 완료																														
개념적 설계	엔티티 Entity	db에 표현하려고 하는 현실 세계의 대상체 파일 시스템에 레코드에 대응 독립적인 의미를 갖는다.																														
	속성 Attribute	개체의 성질, 분류, 식별, 수량 상태 데이터의 가장 작은 논리적 단위 독립적인 의미를 갖지 않음																														
	관계 Relationship	두 개체 간의 의미 있는 연결 관계로 연결되어 있는 개체 타입들의 개수를 <u>관계의 차수</u> 라 부른다. 