1과목

소프트웨어 설계

요구사항 개발 프로세스 : 도출 → 분석 → 명세 → 확인  
요구사항 도출 단계 주요 기법은 인터뷰, 설문, 브레인스토밍 등이 있어 청취와 인터뷰 질문 기술이 필요함.

명백한 역할을 가지고 독립적으로 존재할 수 있는 시스템의 부분으로 넓은 의미에서는 재사용되는 모든 단위라고 볼 수 있으며, 인터페이스를 통해서만 접근할 수 있는 것은? Component

요구 사항 명세 기법에 대한 설명

정형 명세법  
- 수학적 기반/모델링 기반  
- Z, VDM, Petri-Net(모형기반)  
- CSP, CCS, LOTOS(대수적방법)  
- 시스템 요구특성이 정확하고 명세가 간결하다. 명세와 구현이 일치.  
- 그러나 이해도가 낮으며 이해관계자의 작성 부담 가중.  
  
비정형명세  
- 상태, 기능, 객체 중심 명세법  
- FSM(Finite state machine)  
- Decision Table, ER모델링  
- State chart(SADT)  
- UseCase : 사용자기반모델링  
- 명세 작성이 간편하고 의사전달 방법이 다양하다.  
- 불충분한 명세가능성, 모호성.

럼바우(Rumbaugh) 분석기법

1. 객체(Object) 모델링 : 정보모델링, 시스템에서 요구되는 객체를 찾아내어 속성과 연산 식별 및 객체들 간의 관계를 규정, 객체 다이어그램으로 표시  
2. 동적(Dynamic) 모델링 : 상태도(상태 다이어그램)을 이용하여 시스템의 행위를 기술  
3. 기능(Functional) 모델링 : 자료 흐름도를 이용하여 다수의 프로세스들 간의 자료 흐름을 중심으로 처리 과정 표현

소프트웨어를 개발하기 위한 업무(비즈니스)를 객체, 속성 등의 개별요소로 추상화 하는 기법을 "객체지향 분석"이라고 합니다.

HIPO(Hierarchy Input Process Output)

하향식 소프트웨어 개발을 위한 문서화 도구

HIPO 차트 종류에는 가시적 도표, 총체적 도표, 세부적 도표

기능과 자료의 의존 관계를 동시에 표현 가능

보기 쉽고 이해하기 쉽다

소프트웨어 아키텍처 설계에서 시스템 품질속성

가용성, 변경용이성, 성능, 보안성, 사용편의성, 시험용의성 6가지가 있다.

CASE(Computer-Aided Software Engineering)의 원천 기술

구조적 기법, 프로토타이핑 기술, 자동프로그래밍 기술, 정보 저장소 기술, 분산 처리 기술  
이 있다.

|  |  |
| --- | --- |
|  | Usecase 구성요소와의 관계 - 연관 : use case 와 actor의 관계 - 확장 : 기본 use case 수행 시 특별한 조건을 만족할 때 수행할 usecase - 포함 : 시스템의 기능이 별도의 기능을 포함 - 일반화 : 하위 use case/action이 상위 use case/actor에게 기능/역할을 상속받음 - 그룹화 : 여러개의 usecase를 단순화하는 방법 |

명백한 역할을 가지고 독립적으로 존재할 수 있는 시스템의 부분으로 넓은 의미에서는 재사용되는 모든 단위라고 볼 수 있으며, 인터페이스를 통해서만 접근할 수 있는 것은? Component(요소)

**GoF(Gang of Four)**

[구조패턴]  
-어댑터 -브릿지 -컴포지트 -데코레이터 -퍼싸트 -플라이웨이트 -프록시

CASE(Computer-Aided Software Engineering)도구에 대한 설명

소프트웨어 공학의 자동화를 의미, 소프트웨어 공학작업을 자동화한 소프트웨어 패키지를 CASE도구라고 한다.  
CASE도구들은 소프트웨어 관리자들과 실무자들이 소프트웨어 프로세스와 관련된 활동을 지원한다. 즉, 프로젝트 관리 활동을 자동화하고, 결과물을 관리하며, 엔지니어들의 분석, 설계 및 코딩과 테스트작업을 도운다.

주요기능: 다양한 소프트웨어 개발 모형 지원, 그래픽 지원, 소프트웨어 생명주기의 전단계 연결

1.소프트웨어 개발 과정의 일부 또는 전체를 자동화하기 위한 도구이다.

2.표준화된 개발 환경 구축 및 문서 자동화 기능을 제공한다.

3.작업 과정 및 데이터 공유를 통해 작업자간 커뮤니케이션을 증대한다.

4.~~2000년대 이후 소개되었으며, 객체지향 시스템에 한해 효과적으로 활용된다.~~

1980년대 소개, 1990년대부터 자주 사용, 객체지향 시스템 및 구조시스템등 모든분야 적용

3과목

<문제 해설>  
- 후보키(Candidate Key) : 릴레이션을 구성하는 속성 중 튜플을 유일하게 식별하기 위한 기본키로 사용할 수 있는 속성들을 뜻한다.  
  
- 기본키(Primary Key) : 후보키 중에서 선택된 주키(Main Key)이다.  
  
- 슈퍼키(Super Key) : 한 릴레이션 내에 있는 속성들의 집합으로 구성된 키이다. 슈퍼키는 릴레이션에 있는 튜플에 대해 유일성을 만족시키지만, 최소성은 만족 시키지 못한다.  
  
- 외래키(Foregin Key): 관계를 맺고 있는 릴레이션 R1, R2에서 릴레이션 R1이 참조하고 있는 릴레이션 R2의 기본키와 같은 R1릴레이션의 속성 말한다.

- INSERT는 테이블에 데이터를 입력할 때 사용 된다.  
- DROP는 테이블에서 컬럼을 삭제할 수 있으며, 데이터도 함께 삭제 된다. 한번에 하나의 컬럼만 삭제 가능하다.  
- MODIFY는 테이블에 존재하는 컬럼의 데이터 유형, 기본값(default), NOT NULL 제약조건에 대해 변경할 때 사용한다.  
<문제 해설>  
DCL(Data Control Language) : 데이터베이스 관리자(DBA)가 데이터 관리를 목적으로 사용되는 명령어를 뜻함  
여기서, 명령어는 4가지로  
1. GRANT: 데이터베이스 사용자에게 사용권한을 부여하는 명령어  
2. REVOKE: 데이터베이스 사용자의 사용권한을 취소하는 명령어  
3. COMMIT: 데이터베이스 조작 작업이 정상적으로 완료 되었음을 알려주는 명령어  
4. ROLLBACK: 데이터베이스 조작 작업이 비정상적으로 작동하여 원래 상태로 복구하는 명령어

DDL(데이터 정의어): CREATE, ALTER, DROP, TRUNCATE  
DML(데이터 조작어): INSERT, UPDATE, DELETE, SELECT  
DCL(데이터 제어어): GRANT, REVOKE  
TCL(트랜잭션 제어어): COMMIT, ROLLBACK, CHECKPOINT

<문제 해설>  
- 먼저 첫번째 select 과목이름 From 성적 where exists 구문을 살펴 보면, 성적 테이블에서 과목이름을 찾으라고 되어 있고, exists는 in과 비슷한 역할을 한다.  
  두번째 Select ~학생.주소 구문을 살펴 보면, 학생 테이블의 학번과 성적 테이블의 학번은 서로 동일하고, 학생 테이블에서 학과는 전기 또는 전산이고 주소가 경기인 학생을 찾는 구문이다.  
  
여기에 해당되는 정보는 학번 2000, 4000 학생으로 성적 테이블에서 학번을 찾아보면 여기에 해당되는 과목은 "DB, DB, 운영체제" 이다.

<문제 해설>  
관계대수 : 관계형 데이터베이스에서 원하는 정보와 그 정보를 검색하기 위해서 어떻게 유도하는가를 기술하는 절차적인 언어이다.  
비절차적 특성은 관계해석이 지니고 있다.

<문제 해설>  
차수는 +를 해주고 카디널리티는 \*를 해주면 해당 결과가 나옵니다.  
차수 = 4+6=10  
카디널리티 = 5\*7=35  
차수 = 열  
카디널리티 = 행

<문제 해설>  
1NF : 도메인이 원자값만으로 구성된 정규형 \*  
2NF : 기본 키가 아닌 모든 속성이 기본 키에 대해 완전 함수적 종속을 만족하는 정규형  
3NF : 기본 키가 아닌 모든 속성이 기본 키에 대해 이행적 종속을 만족 안하는 정규형  
BCNF : 모든 결정자가 후보 키인 정규형  
4NF : 다치 종속 A->B가 성립하는 경우 A에 함수적 종속 관계를 만족하는 정규형  
5NF : 모든 조인 종속이 후보 키를 통해서만 성립되는 정규형

<문제 해설>

로킹단위↑ : 로크 수↓, 병행성↓, 오버헤드↓, 공유도↓, 제어기법 간단하여 관리 수월  
로킹단위↓ : 로크 수↑, 병행성↑, 오버헤드↑, 공유도↑, 제어기법 까다로워 관리 복잡

병행제어 로킹(Locking)  
- 데이터베이스 병행 제어를 위해 트랜잭션(transaction)이 접근하고자 하는 데이터를 잠가(lock) 다른 트랜잭션이 접근하지 못하도록 하는 병행 제어 기법이다.  
- 트랜잭션이 어떤 데이터에 접근하고자 할 때 로킹을 수행  
- 로킹이 되어 있는 데이터에는 다른 트랜잭션이 접근할 수 없음  
- 트랜잭션은 로킹이 된 데이터에 대해서만 연산을 수행  
- 로킹 단위: 필드, 레코드, 테이블, 파일, 데이터베이스 모두 로킹 단위가 될 수 있다.  
- 한 번에 로킹 할 수 있는 객체의 크기를 로킹 단위라고 한다.

<문제 해설>  
1NF 조건 : 원자값으로 구성

1NF만족하는 모든 도메인이 원자 값이어야 한다.  
2NF 조건 : 부분 함수 종속 제거 (완전 함수적 종속 관계)

1NF를 만족하고 키가 아닌 모든 속성이 기본키에 대하여 완전 함수적 종속 관계를 만족해야 한다.  
3NF 조건 : 이행 함수 종속 제거

1NF를 만족하고 키가 아닌 모든 애트리뷰트들이 기본 키에 이행적으로 함수 종속되지 않아야 한다.  
BCNF 조건 : 결정자 함수 종속  
4NF 조건 : 다중값(다치) 종속성 제거.

1NF를 만족하고 다치 종속이 제거되어야 한다.  
5NF 조건 : 조인 종속성 제거

<문제 해설>  
분산데이터베이스 투명성 조건  
분산데이터베이스는 6가지 투명성(Transparency)을 만족해야 한다.  
  
1. 분할 투명성 : 하나의 논리적 Relation이 여러 단편으로 분할되어 각 단편의 사본이 여러 site에 저장  
2. 위치 투명성 : 사용하려는 데이터의 저장 장소 명시 불필요. 위치정보가 시스템 카탈로그에 유지되어야 함  
3. 지역사상 투명성 : 지역DBMS와 물리적 DB사이의 Mapping 보장. 각 지역시스템 이름과 무관한 이름 사용 가능  
4. 중복 투명성 : DB 객체가 여러 site에 중복 되어 있는지 알 필요가 없는 성질  
5. 장애 투명성 : 구성요소(DBMS 등)의 장애에 무관한 트랜잭션의 원자성 유지  
6. 병행 투명성 : 다수 트랜잭션 동시 수행시 결과의 일관성 유지, Time Stamp, 2PL 구현  
[해설작성자 : 오타는 많은 혼란을 줍니다]  
  
\* 분산 데이터베이스의 목표  
1. Location Transparency (위치 투명성)  
: 데이터베이스의 실제 위치를 알 필요 없이 단지 데이터베이스의 논리적 명칭만으로 액세스할 수 있음  
2. Replication Transparency (중복 투명성; 복제 투명성)  
: 동일 데이터가 여러 곳에 중복되어 있더라도 사용자는 마치 하나의 데이터만 존재하는 것처럼 사용하고, 시스템은 자동으로 여러 자료에 대한 작업을 수행  
3. Failure Transparency (장애 투명성)  
: 장애가 발생해도 트랜잭션을 정확하게 처리하고 데이터 무결성을 보장함  
4. Concurrency Transparency (병행 투명성)  
: 다수의 트랜잭션들이 동시에 실현되더라도 그 트랜잭션의 결과는 영향을 받지 않음  
5. Division Transparency (분할 투명성)  
: 하나의 논리적 릴레이션이 여러 단편으로 분할되어 각 단편의 사본이 여러 시스템에 저장되어 있음을 인식할 필요가 없음

<문제 해설>  
뷰(View) : 하나 이상의 기본 테이블로부터 유도되어 만들어지는 가상 테이블 / 튜플 : 관계데이터 구조 테이블에서 행을 의미한다. / 카디널리지 : 튜플의 갯수를 의미한다.

트랜잭션 : 데이터베이스에서 하나의 논리적 기능을 수행하기 위한 작업의 단위 또는 한꺼번에 모두 수행되어야 할 일련의 연산

<문제 해설>

관계대수 연산에는  
1. 일반 집합 연산자  
합집합  
교집합  
차집합  
카티션 프로덕트  
2. 순수 관계 연산자  
셀렉트  
프로젝트  
조인  
디비전

<문제 해설>  
SELECT σ시그마  
PROJECT π파이  
JOIN ▷◁ 나비넥타이  
DIVISION ÷나누기  
[해설작성자 : 전자공학도]