

컴퓨터 활용 능력 시험 1급 필기

빈칸 채우기 노트

3

과목 III

데이터베이스

한방에
컴활 1급
끝내기

한방에 컴활 끝내기

핵심 요약 노트

[주의사항] 반드시 읽어주세요!

이 요약노트는 저작권법 제 53조에 의거, 공식적인 저작권이 등록된 저작물입니다. 그러므로 이 요약노트를 무단으로 복사, 재배포, 재판매, 재구매, 작품 일부를 무작위로 도용하는 행위에 대해서는 저작권법 136조에 의거하여 5년 이하의 징역 또는 5천만원 이하의 벌금에 처할 수 있음을 꼭 알아주시길 바랍니다.

이 요약노트에는 개인마다 고유하게 부여된 코드가 눈에 보이지 않는 워터마크로 처리되어 있습니다. 그러므로, 노트의 일부분을 캡처하여 배포하는 경우에도 발견 즉시 노트의 최초 구매자와 배포자를 특정할 수 있음을 알려드립니다.

만약 이 요약노트의 무단 복제 또는 배포 사례를 발견하신 경우, 저희 공식 이메일 주소인 odap.history@gmail.com 으로 제보해 주시기 바랍니다. 해당 배포 사례의 법적 처리가 완료된 후에, 제보자님께는 소정의 사례를 보내 드리겠습니다.

데이터베이스

데이터베이스 개념

데이터베이스 관리 시스템(DBMS)

- **데이터베이스** 관리 시스템, DataBase Management System
- 목적 : 데이터를 **□□에서 □□화**하여 관리함으로서 다음의 목적을 달성
 - 데이터의 **□□성** 유지
 - 데이터의 **□□성** 유지
 - 데이터의 **□□** 보장
 - 데이터의 **□□과 □□성 감소**
- 단점(비용)
 - 하드웨어/DBMS **구입 비용**
 - 기존 데이터의 **전산화 비용**
 - **인적 비용**: DBMS와 데이터베이스 언어 조작 가능한 **고급 프로그래머**
 - **□□비용**: **□□과 복구**에 많은 비용, 시간 소요
- **RDBMS: □□형 데이터베이스**
 - **Relationship**(□□, 행과 열)로 데이터를 구조화하여 관리
 - 종류: MS-SQL Server, ORACLE(오라클), MY-SQL

데이터베이스의 종류

종류	설명
계층적(Hierarchical) 데이터베이스	<ul style="list-style-type: none"> • 트리형 논리구조. 모든 데이터 = □□/□□ 관계 • 검색속도 □□ / 유연성(□↑□↓) / 1:1 또는 1:n 데이터관계 / 레코드 삭제시 자식도 삭제
네트워크형(Network) 데이터베이스	<ul style="list-style-type: none"> • 모든 데이터는 1:n 또는 n:m 데이터관계 • 한 레코드는 여러 레코드와 □□됨/유연성(□↑□↓) / 확장성↓
관계형(Relational) 데이터베이스	<ul style="list-style-type: none"> • 모든 데이터를 개체, 속성, 관계로 구성된 □□□□(□□□□)로 표현 • 1:1, 1:n, n:m 관계 표현

데이터베이스의 설계

1. 데이터베이스 설계 과정

- 데이터베이스 기획 - 요구분석 - 개념설계 - 논리설계 - 물리설계 - 구현 순
 - 1) 데이터베이스 **기획**: 데이터베이스 설계의 시작을 위해 **기획**
 - 2) **요구분석**: □□□□와 **사용□□**을 정의
 - 3) **개념설계**: 정보를 구조화 하기 위해 □-□□□□□□□□을 이용해 **□□적 개념**으로 표현
 - 4) **논리설계**: 데이터모델링, **특정 DBMS의 □□적 자료 구조**로 변환. 여기서 □□□□을 **설계**하고 □□□□를 수행
 - 5) **물리설계**: 논리적 구조로 작성된 데이터를 **□□적 구조의 데이터**로 변환. 저장 레코드의 크기, 순서, 인덱스 구조, 저장 구조 등을 결정
 - 6) **구현**: 설계한 데이터베이스의 **구조(□□□□)**를 실제 존재하는 □□로 **생성**

2. E-R(Entity-Relationship) 모델링(개체-관계 모델링)

- **개체(Entity)-관계(Relationship)-속성(Attribute)**로 데이터를 표현하는 데이터 모델

기호	의미	기호	의미
사각형	□□ 타입	마름모	□□ 타입
타원	□□ 타입	밀줄 타원	□ 속성
실선	□□자		

3. 관계형 모델

- 모든 데이터를 테이블에 저장. 각각의 열 = 속성/행 = 튜플

종류	설명
릴레이션	Relation. 데이터들을 □의 형태로 표현한 것(SQL의 □□□□)
속성	Attribute. 테이블의 한 □(□□) . 개체의 상태/특성을 표현함. 데이터베이스에서 가장 작은 논리적 단위(SQL의 column)
튜플	Tuple. 테이블의 한 □ 으로 하나의 개체인 동시에 여러 속성들의 집합. 각 튜플들은 □□해야 하며 순서에 제한 □ (SQL의 row)
차수	Degree. 테이블이 가진 □□의 갯수 .
기수	Cardinality. 테이블이 가진 □□의 갯수 .
도메인	Domain. 속성(Attribute) 이 취할 수 있는 □□□□의 집합 . 예: '학년'속성 도메인=1,2,3,4/'휴학여부'속성 도메인=예/아니오

4. 데이터 모델 관계

관계	설명
일대일(1:1)	두 테이블 간 각각 □□의 레코드 가 대응. 두 테이블의 연결 필드 가 □□□□□□ 인덱스 로 설정한 경우 예: 주민번호 - 한 주민의 정보
일대다(1:n)	한 테이블의 레코드 하나 에 다른 테이블의 □□레코드 가 대응 예: 부서명 - 부서 내의 여러 사원의 정보
다대다(n:n)	두 테이블 간 여러 레코드가 서로 대응 엑세스에서는 지원 [□□□□]

5. 데이터베이스의 정규화

- 목적 : **속성(Attribute)**들간 □□(□□)성으로 발생하는 **충돌(이상현상)** 제거
데이터 중복/속성들 사이의 □□성 **최소화** → 테이블의 **불일치위험 최소화**
+ 데이터구조 **안정성 최대화**
- ※데이터 **중복성**: **같은 데이터를 □□ 개 가지고 있는 것**. 중복성이 높아지면 데이터의 □□성(□□성)을 유지하기 힘들어지고, 추가적인 비용이 발생함
- 방식 : **중복되는 값**을 보다 **단순한 구조**의(더 적은 수의 필드를 갖는) 테이블로 분리 + 테이블 속성들 사이의 종속성을 배제
- 효과 :
 - 1) 데이터의 삽입/삭제/갱신 이상의 발생 방지
 - 2) 데이터 삽입시 테이블 재구성의 필요성 감소
- 제1정규형 ~ 제□정규형까지 있음

6. 데이터베이스 스키마

- 데이터베이스의 구조를 보여주는 일종의 설계도
 - **□□스키마**: 데이터베이스 전체의 구조를 보여주는 스키마. 데이터베이스 하나에 **단 하나의 스키마**만 존재
 - **□□스키마**: 데이터베이스의 **물리적 저장 구조**를 정의한 스키마
 - **□□스키마(□□스키마)**: 데이터를 **사용자에게 어떻게 보여줄지**를 기술한 스키마

데이터베이스 언어

- 데이터 **□□어(DML, Data Manipulate Language)**: 데이터를 실질적으로 **조회** 또는 **처리**하기 위해 사용
 - **INSERT**(새로운 데이터 **추가**), **UPDATE**(데이터 **수정**), **DELETE**(데이터 **삭제**), **SELECT**(데이터 **조회**)
 - 주로 사용자단에서 사용됨
- 데이터 **□□어(DCL, Data Control Language)**: **데이터 관리**를 위해 사용
 - 데이터 관리를 위해 사용하는 언어
 - 주로 **데이터베이스 관리자**가 데이터의 관리를 목적으로 사용
 - 데이터의 무결성 유지, 데이터 회복, 보안 유지, 병행수행 등을 제어
 - **COMMIT, ROLLBACK, GRANT, REVOKE** 등
- 데이터 **□□어(DDL, Data Definition Language)**: 데이터 **정의**를 위해 사용
 - **데이터베이스를 생성** 및 수정하는 언어
 - 주로 **데이터베이스 개발자**가 데이터베이스 구축을 위해 사용
 - 데이터베이스의 논리, 물리적 구조 정의
 - **CREATE, ALTER, DROP** 등

테이블 활용

테이블 만들기

1. 이름 작성 규칙

- 테이블/필드의 이름 작성 규칙은 다음을 따른다
 - 최대 글자 수는 32자까지 가능
 - 공백 제외한 모든 특수문자 사용가능. 공백 사용 [가 가능]능
 - 이름의 첫 문자는 0이 올 수 없음
 - 하나의 테이블 안에서 각 필드의 이름은 00해야 함(중복 필드명 허용 X)

2. 데이터 형식

- 짧은 텍스트 형식: 최대 255자. 텍스트, 또는 텍스트+숫자 조합.
- 긴 텍스트 형식(메모 형식): 최대 65,535자. 텍스트, 또는 텍스트+숫자 조합
- 숫자 형식: 숫자 데이터 형식. 산술 계산 가능. 기본은 00형(Long)

바이트형(Byte)	1Byte	정수형(Integer)	2Byte
정수형(Long)	4Byte, 소수점 아래 7자리까지 표현		
실수형(Double)	8Byte, 소수점 아래 15자리까지 표현		

- 날짜/시간 형식: 날짜와 시간을 표시. 8Byte(바이트). 조건식에서 쓸 경우 쌍 기호(#)로 감싸야 함.(큰따옴표(“ ”) 아님)
- 통화 형식: 화폐, 재무 계산을 위한 숫자 정보. 8Byte. 소수점 아래 4자리.
- Yes/No 형식: 예/아니오 둘 중 하나만 입력. 0Bit(비트). 예=-1, 아니오=0
- 일련번호 형식: 4Byte. 새 레코드를 만들때 00으로 번호를 1증가시켜 저장하는 형식. 한 번 데이터를 입력한 후에는 업데이트 [가 가능]능. 한 번 필드에 데이터 입력하면 데이터 형식을 0000로 변경 불가.
- OLE 개체 형식: 다른 프로그램(워드, 엑셀 등)에서 만들어진 개체 입력.
- 하이퍼링크 형식: 웹 또는 파일탐색기의 특정한 위치로 이동할 수 있는 하이퍼링크
- 조회방법사 형식: 목록 상자 또는 콤보 상자를 이용해 다른 테이블 값/목록 저장

3. 필드 속성

- 입력마스크: 데이터 입력 시 입력 틀을 만드는 속성

기호	설명	필수	기호	설명	필수
0	숫자 입력(0-9)(공백x)	0	A	영문자/한글/숫자 입력	0
9	숫자 입력(0-9)(공백o)		a	영문자/한글/숫자 입력	
#	숫자, 덧셈셈 기호(+/-)		&	모든 문자/공백 입력	0
L	영문자/한글 입력	0	C	모든 문자/공백 입력	
?	영문자/한글 입력		<	모든 문자→소문자로 변환	
\	모든 문자를 그대로 표시 (예: \# 은 #으로 표시)		>	모든 문자→대문자로 변환	

- 일반 속성: 필드의 형식, 입력 속성을 지정해 줌

형식	데이터 표시 형식 지정
소수 자릿수	데이터의 00점 아래 자리의 수를 지정
캡션	필드명 대신 표시되는 00을 지정
기본값	새로운 레코드 생성시 해당 필드에 00 입력되는 값 지정
필수	0000 입력해야 하는 필드인지 여부 지정
빈 문자열 허용	문자열 입력시 0000(길이=0) 입력 가능 여부 지정
유효성 검사 규칙	필드에 입력되는 데이터의 규칙을 지정 =(일치), <>(불일치), <=(크거나 같음), <(کم) Or(또는), And(이며), like(문자열 포함) in(a,b): a,또는 b중 하나만 가능
유효성 검사 텍스트	유효성 검사 규칙 위배시 표시할 00 0000 지정
IME 모드	IME 모드 여부 지정

- 조회 속성: 사용자가 직접 값을 입력하며 발생하는 오류 방지
 - 다른 테이블/쿼리의 값 목록 또는 자신이 직접 원하는 값의 00 생성 가능
 - 조회 목록으로 표시할 열의 00도 변경 가능

컨트롤 표시	입력 목록을 보여줄 컨트롤 - 00 상자, 00 상자 중 택 1
행 원본 유형	행 원본의 유형 지정. 0000/00, 0 목록(직접 입력), 0 목록 선택 가능
행 원본	행 00 지정. 테이블/쿼리 선택, 값 목록 선택(직접 입력), 필드 목록 선택
바운드 열	선택한 여러 열 중에서 해당 0000에 저장되는 열 지정
열 개수	표시되는 열의 개수 지정
열 이름	열 이름의 표시 여부 지정
열 너비	열의 너비 지정. 여러 개의 열이면 0000(0)으로 구분
행 수	00상자의 행의 갯수 지정
목록 너비	00상자의 목록 너비를 지정
목록 값만 허용	지정한 목록값 00의 값 선택 가능 여부 지정
여러 값 허용	여러 개의 값 선택 00 여부 지정
값 목록 편집 허용	지정한 값을 00할 수 있는지 여부 지정
목록 항목 편집 폼	목록 항목 편집시 실행할 0 지정
행 원본 값만 표시	행 원본과 00하는 값만 표시할 지 여부 지정

키와 인덱스

1. 기본키

- 테이블 내 모든 레코드들을 00하게 식별하기 위한 필드
- 기본 키 필드의 모든 값은 서로 00해야 하며 빈 값이 00어야 함(NOT NULL)
- 단일 필드(하나의 필드) / 다중 필드(여러 필드 조합) / 00번호 기본키(데이터 삽입시 자동으로 00번호 부여 - 마지막으로 생성된 키 +1)
- 반드시 기본키를 설정할 필요 00있 00없음
- 이미 데이터가 입력된 필드도 기본키로 지정 00가 00불가]능. 단, 모든 값들이 각각 00하며 빈 값이 00어야 함
- OLE개체, 첨부파일 형식 필드는 기본키로 지정 00가 00불가]능

2. 인덱스

- 테이블 레코드의 00(검색과 정렬) 속도 향상을 위해 필드에 지정
- 인덱스 지정시 0기(검색/정렬)속도 ↑, 0기(데이터 추가/수정/삭제)속도 ↓
- 여러 필드에 설정 가능. 00을 자주 하거나 00 기준이 되는 필드에 지정함
- 기본키와 마찬가지로 OLE개체, 첨부파일 형식 필드에는 지정 00가 00불가]능

3. 외래키

- 두 0000 간의 관계를 설정하기 위해 지정하는 키
- 두 0000에 공통으로 들어있는 키
- 테이블 간에 외래키로 관계가 설정된 경우, 기본키를 제거하려면 먼저 00를 삭제해야함
- 테이블 A:B가 1:1 관계인 경우: A의 기본키를 B의 외래키로 사용 - A와 B 모두 00한 값만 있어야 함
- 테이블 A:B가 1:n 관계인 경우: A의 기본키를 B의 외래키로 사용 - A에는 고 유한 값만 있어야 하지만 B에는 00된 값 존재 가능

관계 설정

1. 개념

- 테이블 간의 관계 설정. 여러 테이블에 분산된 정보들을 연결할 수 있으며, 데이터의 무결성을 향상시킬 수 있음
- 관계 설정을 하여, 데이터 입력시 조회속성 등을 이용한 올바른 데이터 입력을 보장할 수 있음

2. 방법

- 기본 테이블의 기본키 필드 - 관계 테이블의 00키 필드를 대응시킴
- 기본키 필드와 외래키 필드의 데이터 형식은 00아야 함
- 외래키는 기본 테이블과 관계 테이블에 00 존재해야 함
- 00 00성: 참조하는 테이블의 외래키와 연결된 기본키가 반드시 참조되는 테이블에 존재해야 함. 즉, 외래키는 항상 기본 테이블의 기본키 필드의 값들과 동일하도록 유지함
- 테이블 관계 편집을 위해서는 테이블을 00 다음 진행해야 함

3. 관계 편집 대화상자

- **항상** ☐ **성 유지**: 기본 테이블에 존재하지 **않는** 키를 관계 테이블에 생성할 수 ☐ **도록 강제함**
- **관련 필드 모두 업데이트**: 기본 테이블의 **기본 키 값이 바뀌면** 연결된 테이블의 해당 필드의 값도 **함께** ☐ **으로** ☐ **되도록 설정**
- **관련 레코드 모두 삭제**: 기본 테이블의 **레코드 삭제**시 관련 테이블의 **연결된 레코드들도** ☐ **으로** ☐ **되도록 설정**

4. 관계의 종류

- **일대일**: 기본 테이블의 기본키와 연결된 외래키가 **관계 테이블의** ☐ **키임**
- **일대다**: 기본 테이블의 기본키와 연결된 외래키가 **관계 테이블의 기본키가** ☐
- **다대다**: 액세스에서 지원 ☐ **함** ☐ **하지 않음**

데이터 내보내기/가져오기/연결하기

- **내보내기**
 - 테이블/쿼리/폼/보고서 → 텍스트, 엑셀, 워드, 액세스 데이터베이스, XML 등으로 내보낼 수 있음
 - **전체 개체 or 데이터**(추가서식x) 내보내기 선택 ☐ **가** ☐ **불가** **능**
 - 쿼리 내보내기: **쿼리의** ☐ ☐ ☐ **가 저장됨**
 - **보고서** 내보내기: **보고서에** ☐ **된 데이터만** 내보냄(원본 테이블 내보내기 x)
 - 내보내기 가능한 파일 형식: ☐ ☐ (xlsx), ☐ ☐ (mdb, accdb), ☐ ☐ (txt), ☐ (xml, html), ☐ ☐ ☐ ☐ 파일(dBASE, ODBC), **PDF** 등
- **가져오기 기능**
 - **텍스트, 엑셀, 액세스 데이터베이스, XML** 파일 가져오기 가능(**워드파일** ☐

종류	설명
엑세스 개체 가져오기	테이블/쿼리/매크로/프로시저/보고서 등 모든 개체 가져오기 가능(같은 프로그램에서 작성하였으므로). 필드는 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> 개까지만 가져오기 가능
테이블 가져오기	테이블 <input type="checkbox"/> <input type="checkbox"/> 도 복사 가능. 테이블 <input type="checkbox"/> <input type="checkbox"/> 만 가져오기 가능(빈 테이블 생성됨)
엑셀 가져오기	한 번에 하나의 워크시트 가져오기 가능. 매크로 가져오기 <input type="checkbox"/>
텍스트 가져오기	텍스트 가져오기 마법사 사용. 단, 가져오는 과정에서 수정은 <input type="checkbox"/> 가 <input type="checkbox"/> 불가 능

- 데이터를 가져와도 **원본 데이터에 영향** ☐ ☐ **x**
- 가져오는 개체의 이름이 이미 있다면, 가져오는 개체 뒤에 1,2,3... 이 추가됨
- **연결하기 기능**
 - **다른** ☐ ☐ ☐ ☐의 데이터를 **엑세스와 연결**
 - 액세스에서 데이터를 수정하면 ☐ ☐ ☐ ☐도 수정됨
 - 단, **연결한 테이블 삭제**시 원본 데이터와 ☐ ☐이 **끊어짐**. 원본 데이터에 **영향을 미** ☐ **침** ☐ **치지 않음**
 - 원본 데이터 삭제시 **연결한 테이블도** ☐ ☐ **됨**
 - 연결 가능한 파일 형식: Access 데이터베이스, Excel 스프레드시트, .txt, HTML 문서, 타사 데이터베이스(dBASE, ODBC), Outlook 등

II 쿼리 활용

조회 쿼리: SELECT문

문법: **SELECT** 필드명 **FROM** 테이블명

- ☐ ☐을 만족하는 데이터를 조회하기 위해 사용하는 쿼리문입니다. 항상 **SELECT** 필드명 **FROM** 테이블명 의 형식을 따릅니다
- **여러 필드**는 ☐ ☐ (☐)로 구분합니다
예시) SELECT 매장명 ☐ 고객수 FROM 매장
- **모든 필드**를 조회하려면 ☐ ☐를 사용합니다
예시) SELECT ☐ FROM 매장
- 여러 테이블에서 조회할 때에는 **테이블명도** ☐ ☐로 구분해 줍니다. 이 경우 필드명은 **‘테이블명 ☐ 필드명’**으로 입력합니다
예시) SELECT 매장 ☐ 매장명 ☐ 직원 ☐ 직원명 FROM 매장 ☐ 직원
- 필드명에 **공백**이 있는 경우 필드명을 ☐ **괄호** (☐)로 묶어줍니다
예시) SELECT ☐ 매출 비중 ☐ ☐ 폐점 여부 ☐ FROM 매장

• 사용 예시

- 1) **SELECT** 매장명 **FROM** 매장
: 매장 테이블에서 매장명 필드 조회
- 2) **SELECT** 매장명 ☐ 매출 **FROM** 매장
: 매장 테이블에서 매장명, 매출 필드 조회
- 3) **SELECT** ☐ **FROM** 매장
: 매장 테이블에서 모든 필드 조회
- 4) **SELECT** 매장 ☐ 매장명 ☐ 직원 ☐ 이름 FROM 매장 ☐ 직원
: 매장 테이블에서 매장명 필드를, 직원 테이블에서 이름 필드를 조회

레코드 검색: WHERE절

문법: **SELECT** 필드명 **FROM** 테이블명 **WHERE** 조건

- 필드 중에서 **특정** ☐ ☐을 만족하는 레코드를 검색합니다.
- 숫자, 텍스트, 논리값 등을 조건으로 지정할 수 있습니다.
- **텍스트**를 조건으로 지정할 때는 ☐ ☐ **표** (☐)로 감싸줘야 합니다
- 사용 예시: ‘매장’ 테이블에서 조회시
 - 1) SELECT * FROM 매장 **WHERE** 고객수 ☐ **1000**
: ‘매장’ 테이블에서 **‘고객수’가 1000 이상인** 레코드 검색
 - 2) SELECT * FROM 매장 **WHERE** 폐점여부 ☐ **TRUE**
: ‘매장’ 테이블에서 **‘폐점여부’가 참인** 레코드 검색
 - 3) SELECT * FROM 매장 **WHERE** 매장명 ☐ **일산점** ☐
: ‘매장’ 테이블에서 **‘매장명’이 “일산점”인** 레코드 검색

문자열 포함: LIKE

문법: **SELECT** ~ **FROM** ~ **WHERE** ~ **LIKE** “문자열”

- 주로 SELECT ~ FROM ~ WHERE 문과 함께 쓰이며, **LIKE** 뒤에 오는 “문자열”과 ☐ ☐ 또는 ☐ ☐한 레코드를 찾기 위해 사용합니다.
- 와일드카드: ☐ (**길이 상관 x**) / ☐ (**길이 상관 o**)
- 조건 패턴: **범위(시작문자 ☐ 끝문자)**, **제외(☐)**
- 예시: 매장 테이블에서 LIKE를 사용하여 조회
SELECT * FROM 매장 WHERE 매장명 **LIKE** “강남점”
→ 매장명이 “강남점”인 레코드 모두 조회(**정확히** ☐ ☐해야함)
- 2) SELECT * FROM 매장 WHERE 매장명 **LIKE** “강”
→ 매장명이 “강”으로 ☐ ☐하는 레코드 모두 조회
- 3) SELECT * FROM 매장 WHERE 매장명 **LIKE** “*강”
→ 매장명이 “강”으로 ☐ ☐는 레코드 모두 조회
- 4) SELECT * FROM 매장 WHERE 매장명 **LIKE** “*강*”
→ 매장명에 “강”이 ☐ ☐된 레코드 모두 조회(**위치, 길이** ☐ ☐)
- 5) SELECT * FROM 매장 WHERE 매장명 **LIKE** “강??”
→ 매장명이 “강”으로 ☐ ☐하는 ☐ 글자인 레코드 모두 조회
- 6) SELECT * FROM 매장 WHERE 매장명 **LIKE** “[ㄱ-ㄷ]”
→ 매장명이 ☐ ☐ ☐으로 **시작**하는 레코드 모두 조회

논리 연산자, 비교 연산자

- **AND**: a AND b → a와 b가 ☐ ☐ **참**이어야 참
예) >=5 AND <=10: 5 이상이며 10 이하일 때 참
- **OR**: a OR b → a와 b 중 ☐ ☐ ☐ **참**이어도 참
예) “서울” OR “경기”: 서울 또는 경기일 때 참
- **NOT**: NOT a → a가 ☐ ☐ **일 경우 참**
예) NOT >=5: 5 이상이 ‘아닐’ 경우 참(5 미만)
- **비교연산자**
 - ① = (☐) ② <> (☐) ③ > (☐, ☐) ④ < (☐, ☐)
 - ⑤ >= (☐거나 ☐, ☐) ⑥ <= (☐거나 ☐, ☐)
- **IS**: Null 체크를 위해 사용. IS Null → Null인 경우에 ☐
※논리/비교 연산자는 주로 **SELECT ~ FROM ~ WHERE** 결과 **함께** 쓰이며 WHERE 뒤에 오는 조건 설정에 사용됩니다
예: SELECT * FROM 매장 WHERE 고객수 >=100 AND 고객수 <= 200 → 매장 테이블에서 고객수가 100 이상이며 200 이하인 레코드를 조회

산술연산자

40 • +: ☐ ☐ 연산. 2+3 = 5

- -: □□ 연산. 3-2=1
- ^: □□ 연산. 3^2=9
- *: □□ 연산. 3*2=6
- MOD: □□□ 구하기. 9 MOD 2=1
- &: 문자열 □□기. “대한” & “민국”= “대한민국” / “1” & “2”= “12”

범위 안에 들어오는지 확인: BETWEEN

문법: SELECT 필드명 FROM 테이블명 WHERE 조건
BETWEEN 범위값1 AND 범위값2

- 주로 WHERE절과 함께 사용하며, 조건의 값이 범위값1과 범위값2 □□에 들어오는 레코드를 조회합니다
- 예시: SELECT * FROM 매장 WHERE 고객수 BETWEEN 500 AND 1000
→ 매장 테이블에서 고객수 값이 □□~□□□□ □□에 들어오는 레코드를 조회

인수 목록으로 확인: IN

문법: SELECT 필드명 FROM 테이블명 WHERE 조건
IN(값1, 값2, ...)

- 주로 WHERE절과 함께 사용하며, IN 뒤에 오는 □□에 들어있는 값에 해당하는 모든 레코드를 조회합니다
- 예시: SELECT * FROM 매장 WHERE 매장명 IN(‘강남점’, ‘강서점’, ‘인천점’)
→ 매장 테이블에서 매장명 값이 ‘□□□’ 또는 ‘□□□’ 또는 ‘□□□’인 모든 레코드를 조회

중복 제거: DISTINCT

문법: SELECT DISTINCT 필드명 FROM 테이블명 ~

- □□을 제거한 레코드를 조회하기 위해 사용합니다
- 예시: SELECT DISTINCT 매장명 FROM 매장
→ 매장 테이블에서 매장명 값의 □□을 제거한 후 매장명 조회

레코드 정렬: ORDER BY절

문법: SELECT 필드명 FROM 테이블명 ORDER BY 필드명 (ASC 또는 DESC)

- 필드명에 해당하는 값들을 기준으로 □□차순(ASC)/□□차순(DESC)으로 정렬해줍니다.
- SELECT 문의 가장 □□□□에 옵니다.
- (ASC 또는 DESC)는 □□ 가능하며, 생략시 기본값은 ASC(□□차순)입니다
- 예시: ‘매장’ 테이블에서 조회시
 - 1) SELECT * FROM 매장 ORDER BY 매장명
: ‘매장’ 테이블 값을 ‘매장명’ 필드의 값 기준으로 □□차순 정렬. 이 때 순서는 가나다 순을 따릅니다.
 - 2) SELECT * FROM 매장 ORDER BY 매출 ASC
: ‘매장’ 테이블 값을 ‘매출’ 필드의 값 기준으로 □□차순 정렬
 - 3) SELECT * FROM 매장 ORDER BY 매출 DESC
: ‘매장’ 테이블 값을 ‘매출’ 필드의 값 기준으로 □□차순 정렬

두 테이블 통합 검색 쿼리: UNION

문법: SELECT 필드명 FROM 테이블1 UNION SELECT 필드명 FROM 테이블2

- 두 테이블의 레코드를 □□해서 검색하여 보여주는 쿼리
- 필드명은 여러 개가 들어갈 수 있으나, 각 테이블 별로는 같아야 합니다
- UNION: 중복된 데이터는 □□함
- UNION ALL: 중복된 데이터도 □□ 보여줌

두 테이블 연결 검색 쿼리: JOIN

문법: SELECT 필드명 FROM 테이블1 JOIN 테이블2 ON
테이블1.필드명=테이블2.필드명

- 두 테이블의 레코드 중 지정된 □□의 값이 □□ 데이터를 모두 보여줍니다
- 두 테이블의 필드명은 같아야 하며, 필드명에 해당하는 값이 동일한 레코드를 모두 가져옵니다
- INNER JOIN(=JOIN): 기본값. JOIN으로 입력하면 INNER JOIN으로 동작
두 테이블에서 JOIN된 □□의 값이 □은 레코드만 가져옵니다

- LEFT JOIN: 왼쪽 외부 조인. 왼쪽 테이블은 □□ 데이터를, 오른쪽 테이블은 JOIN된 필드의 □이 □은 레코드만 가져옵니다
- RIGHT JOIN: 오른쪽 외부 조인. 오른쪽 테이블은 □□ 데이터를, 왼쪽 테이블은 JOIN된 필드의 값이 □은 레코드만 가져옵니다
- 예시) 왼쪽 = ‘기획팀’ 테이블, 오른쪽 = ‘등산회’ 테이블

‘기획팀’ 테이블		
사번	이름	직급
003	김부장	부장
005	서대리	대리
008	박사원	사원

‘등산회’ 테이블	
사번	이름
003	김부장
004	최부장
010	심차장

SELECT 기획팀.*, 등산회.* FROM 기획팀 JOIN 등산회 ON 기획팀.사번 = 등산회.사번;

기획팀.사번	기획팀.이름	직급	등산회.사번	등산회.이름
003	김부장	부장	003	김부장

SELECT 기획팀.*, 등산회.* FROM 기획팀 LEFT JOIN 등산회 ON 기획팀.사번 = 등산회.사번;

기획팀.사번	기획팀.이름	직급	등산회.사번	등산회.이름
003	김부장	부장	003	김부장
005	서대리	대리		
008	박사원	사원		

SELECT 기획팀.*, 등산회.* FROM 기획팀 RIGHT JOIN 등산회 ON 기획팀.사번 = 등산회.사번;

기획팀.사번	기획팀.이름	직급	등산회.사번	등산회.이름
003	김부장	부장	003	김부장
			004	최부장
			010	심차장

레코드 추가 쿼리: INSERT INTO ~ VALUE

문법: INSERT INTO 테이블명 (필드명1, 필드명2, ...)
VALUE(필드값1, 필드값2, ...)

- 테이블에 레코드를 □□하는 쿼리
- 필드명의 갯수와 필드값의 □□는 항상 같아야 합니다
- 텍스트 또는 날짜 값은 □□□□□□로 감싸줘야 합니다
- 예시: ‘매장’ 테이블에 데이터 추가시
INSERT INTO 매장(매장명, 지역, 매출, 고객수) VALUES (‘일산점’, ‘경기’, 12000000, 1500)
: 매장명은 일산점, 지역은 경기, 매출은 12000000, 고객수는 1500인 데이터를 추가합니다

레코드 삭제 쿼리: DELETE FROM ~

문법: DELETE FROM 테이블명 WHERE 조건

- 테이블에서 조건에 해당하는 레코드를 □□하는 쿼리
- WHERE 조건에 해당하는 레코드만 삭제합니다
- WHERE 조건을 생략하면 □□ 레코드를 삭제합니다
- 한 번 삭제한 레코드는 복구 또는 작업 취소가 □□가 □□불가□□합니다
- 예시: ‘매장’ 테이블에 데이터 삭제시
DELETE FROM 매장 WHERE 매장명=‘일산점’
: 매장명이 일산점인 레코드를 매장 테이블에서 삭제합니다

레코드 업데이트 쿼리: UPDATE ~ SET

문법: UPDATE 테이블명 SET 필드명1=필드값1,
필드명2=필드값2, ... WHERE 조건

- 테이블에서 조건에 해당하는 레코드를 □□하는 쿼리
- WHERE 조건에 해당하는 레코드만 업데이트합니다
- WHERE 조건을 생략하면 □□ 레코드를 업데이트합니다

- 예시: '매장' 테이블의 데이터 업데이트
UPDATE 매장 SET 매출 = 매출*0.8
:매장 테이블에서 000 레코드의 매출 필드 값을 (현재) 매출 * 0.8 값으로 업데이트 합니다
UPDATE 매장 SET 지역='경기북부' WHERE 매장명='일산점'
:매장명이 0000 인 레코드의 지역 필드 값을 00000 로 업데이트합니다

매개변수 쿼리

- 쿼리 실행시 마다 사용자로부터 조건값을 000 받아 실행할 수 있는 쿼리
- 대화상자 표시 키워드: 0000(000) 로 감싼 텍스트를 입력. ‘.’, ‘!’, ‘[’, ‘]’ 등의 특수문자는 입력 [0가0불가]능
- 조건: Like키워드 + 와일드카드 문자(0,0)를 포함한 조건 입력
- 예시(사용자가 입력한 필드보다 큰 값을 검색할 때): >= [최소값 입력]
→ ‘최소값 입력’이라는 텍스트가 포함된 00000 표시. 00000에서 사용자가 입력한 값보다 크거나 같은 값 조회

집단함수

1. 집단함수의 기능

- 각 레코드를 필드명별로 그룹화한 다음 각 그룹에 대한 000을 수행하기 위한 함수

2. 집단함수의 종류

- COUNT: 그룹에 속한 레코드의 000
- SUM: 그룹에 속한 레코드의 특정 필드 값의 000
- AVG: 그룹에 속한 레코드의 특정 필드 값의 000
- MAX/MIN: 그룹에 속한 레코드의 특정 필드 값들의 000/000값

3. GROUP BY ~ HAVING절

- 문법: SELECT 필드명1 FROM 테이블명 GROUP BY 필드명2 HAVING 조건
- GROUP BY 필드명2: 필드명2를 기준으로 000화를 합니다
 - HAVING 조건: 그룹화 한 다음 000에 맞는 000의 데이터만 표시합니다.
WHERE 절과 비슷하지만, 그룹화를 한 다음 조건을 체크한다는 점을 유의해야 합니다
 - 예시) '매장' 테이블 원본

	매장명	고객수	지역
1	강남점	100	서울
2	강서점	150	서울
3	일산점	80	경기
4	수원점	110	경기
5	부산점	120	부산
6	해운대점	90	부산

쿼리 1) SELECT 지역,SUM(고객수) FROM 매장 GROUP BY 지역
→ ①지역필드의 값을 기준으로 그룹화한 다음, ②SUM(고객수)로 고객수의 합을 구하고, ③ SELECT 지역,SUM(고객수) 로 지역과 고객수의 합을 표시합니다
결과 1)

	지역	SUM(고객수)
1	서울	250
2	경기	190
3	부산	210

쿼리 2) SELECT 지역,SUM(고객수) FROM 매장 GROUP BY 지역 HAVING SUM(고객수) >= 200
→ ①지역필드의 값을 기준으로 그룹화한 다음, ②SUM(고객수)로 고객수의 합을 구하고, ③ 그 중 SUM(고객수)가 200이상인 레코드만 골라서 ④ SELECT 지역,SUM(고객수) 로 지역과 고객수의 합을 표시합니다
결과 2)

	지역	SUM(고객수)
1	서울	250
2	부산	210

도메인함수

1. 도메인함수의 기능

- 레코드 집합에 대한 계산을 하는 함수
- 도메인함수의 문법: 도메인함수 (“인수,”도메인,”조건”)
인수 = 000명, 도메인 = 0000명
각 인수는 0000표(00)로 묶어줍니다. 조건 안의 문자열은 00표(00)로 묶어줍니다

3. 도메인함수의 종류

- ① DCOUNT: 필드의 000 ② DSUM: 필드의 000
③ DAVG: 필드의 000 ④ DMAX/DMIN: 필드의 000/000값
⑤ DLOOKUP: 필드의 000
- 예시)
=DSUM(“고객수”, “매장”, “지역=’서울’”) → ‘매장’ 테이블에서 ‘지역’ 필드 값 = ‘서울’인 레코드들의 ‘고객수’ 값의 합계
=DCOUNT(“*”, “학생”, “학년=1”) → ‘학생’ 테이블에서 ‘학년’ 필드값 = 1인 레코드들의 갯수
=DLOOKUP(“성명”, “사원”, “[사원번호]=1”) → ‘사원’ 테이블에서 ‘사원번호’ 필드값 = 1인 레코드의 ‘성명’필드의 값

문자열함수

1. 문자열함수의 종류

- INSTR(문자열, 찾는문자): 문자열에서 찾는 문자의 000
예: = INSTR(“대한민국만세”, “만”) = 0
- LEFT/RIGHT(문자열, 자릿수): 문자열에서 왼쪽(LEFT)/오른쪽(RIGHT) 끝에서부터 000만큼의 문자열
예: = LEFT(“대한민국만세”, 3) = “0000”
= RIGHT(“대한민국만세”, 4) = “00000”
- MID(문자열, 시작위치, 자릿수): 문자열에서 000 위치에서 시작하여 자릿수만큼의 문자열
예: =MID(“대한민국만세”, 2, 3) = “0000”
- STRCOMP(문자열1, 문자열2): 문자열 1과 문자열 2를 비교하여 같으면 0, 다르면 00 반환
예: = STRCOMP(“대한민국”, “대한민국”) = 0
= STRCOMP(“대한민국”, “한국”) = 00

날짜/시간 함수

1. 날짜/시간 함수의 종류

- DATE() / TIME(): 현재 000(DATE)/000(TIME) 표시
- YEAR(날짜)/MONTH/DAY: 날짜에서 0(YEAR)/0(MONTH)/0(DAY)을 표시
- HOUR(시간)/MINUTE/SECOND: 시간에서 0(HOUR)/0(MINUTE)/0(SECOND)를 표시
- DATEADD(형식, 값, 날짜): 날짜에서 형식(연=“YYYY”,월=“M”,일=“D”)을 값만큼 0합
예: =DATEADD(“M”,3,DATE()) = 오늘날짜 + 3 개월(M)
=DATEADD(“D”,10,DATE()) = 오늘날짜 + 10 일(D)
- DATEDIFF(형식, 날짜1, 날짜2): 날짜 1과 2 사이의 000를 형식으로 나타냄
예: =DATEDIFF(“YYYY”, “1990-03-02”, DATE())
= 1990년 3월 2일부터 오늘까지의 년 수(차이)

|| 품과 컨트롤 활용

품의 개념과 종류

1. 품의 개념

- 테이블에 레코드를 000 및 000하는 작업을 위한 인터페이스
- 테이블 레코드, 쿼리 등을 입력 또는 편집

2. 품의 종류

- 언바운드 품: 연결된 데이터(레코드)가 0는 품. 기본 품
- 바운드 품: 연결된 데이터(레코드)가 0는 품. 언바운드 품에 ‘0000 000’ 42 속성을 이용해 레코드를 연결하면 바운드 품이 됨

폼의 구성요소

폼 머리글 /바닥글	모든 레코드에 동일하게 적용되는 정보(폼 제목 등) 폼 보기에서는 □□ 상단(/하단)에 표시. 인쇄할 때에는 1페이지 상단(/하단)에 □□ 표시
페이지 머리글 /바닥글	모든 페이지 상단(/하단)에 동일하게 적용되는 정보 날짜 등. 폼 보기에서는 표시[□□□□]
본문	내용이 표시되는 영역

폼의 속성

1. 형식

캡션	폼 보기 상태에서 □□ 표시줄에 표시될 텍스트 설정
기본 보기	폼을 여는 보기 형식 지정 ① 단일폼: 레코드를 한 번에 □□씩 표시 ② 연속폼: □□ 레코드를 동시에 표시(창을 채울 정도) ③ 데이터시트: □□/□□로 이루어진 □□ 형태로 표시 ④ 분할 표시 폼: 한 화면에 폼 보기와 데이터 시트 보기 로 □□해서 □□에 표시함
자동 크기 조정	폼 창을 열 때 □□를 자동으로 조절하여 레코드를 모두 표시할 수 있도록 할 지 여부 지정
자동 가운데 맞춤	폼을 열 때 창을 자동으로 □□□□ 맞춤 지 여부 지정
컨트롤 상자 최소화/최대화 단추	□□□□ 메뉴(조절 메뉴상자 & 제어상자) 표시 여부 지정 □□화/□□화 버튼 표시 여부 지정
닫기 단추	닫기 버튼 표시 여부 지정
탐색 단추	탐색 단추 표시 여부 지정
레코드 선택기	레코드 선택기 표시 여부 지정
스크롤 막대	스크롤 막대 표시 여부 지정

2. 데이터/기타

레코드 원본	폼의 □□□□□ 지정. □□□□/□□ 등을 지정 가능
필터	데이터 □□□□의 기준 설정
필터사용	필터 여부 지정
정렬 기준	데이터를 □□□□할 기준 설정
편집,추가, 삭제 가능	폼 내용 편집,추가,삭제 □□□□ 여부 설정
레코드잠금	두 명 이상 사용자가 동시에 편집할 때 레코드 □□□□ 여부 설정
모달	폼을 □□□□으로 열 지 여부

폼 작성 - 폼방법사

• 선택 가능한 폼 모양의 종류

컬럼 형식	각 필드가 레이블과 함께 각 □□에 표시됨. 각각의 컨트롤 을 다른 크기로 변경 불가능
테이블 형식	레이블은 폼 상단에 □□□□만 표시되고, 각 레코드의 필드 들이 한 줄에 나타남
데이터시트 형식	□□□□과 동일한 형식으로 표시
맞춤	필드에 들어가는 값들의 크기(양)에 따라 필드를 자동으로 □□□□배분하여 배치

폼 컨트롤 개요

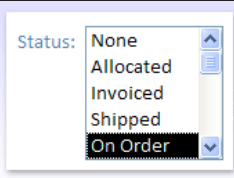
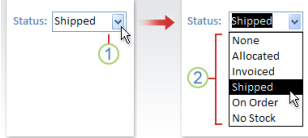
1. 개념

- 폼/보고서에 □□□□를 표시하고 함수/매크로 □□□□을 실행하기 위해 사용하는 GUI(그래픽) 개체.
- 텍스트상자, 콤보상자, 명령 단추 등이 해당

2. 종류

바운드 컨트롤	· 테이블/쿼리의 필드와 컨트롤 □□□□가 연결된(bound) 컨트롤 · 텍스트상자, 콤보상자, 목록상자, 확인란 등 · 연결된 데이터의 표시/입력/수정이 [□□가 □□불가]능 · '□□□□' 속성을 이용하여 바운드
언바운드 컨트롤	· 테이블/쿼리의 필드 - 컨트롤 원본 데이터가 연결되지 □□은 (unbound) 컨트롤 · 명령단추, 레이블 등 · 언바운드 컨트롤 → 바운드 컨트롤로 수정 [□□가 □□불가]능(레 코드 원본 속성 지정)
계산 컨트롤	· 원본 데이터로 계산을 수행하는 □□□□이 지정된 컨트롤 · '□□'을 입력한 다음 식을 지정함. · 값을 직접 입력할 수 [□□있 □□없]음

폼 컨트롤의 요소

텍스트 상자	<ul style="list-style-type: none">· □□□나 계산□□를 표시. 바운드 컨트롤로 사용 [□가 □불가]능· 원본 데이터 표시. □□□된 경우 컨트롤의 값 변경 □ 원본 데이터도 변경· 원본 데이터의 □□□도 표시 가능(=로 시작). 이 경우 컨트롤 값 변경 □ 원본 데이터 값 변경						
목록상자	<ul style="list-style-type: none">· 목록의 데이터 중에서 하나□□을 선택할 수 [□있 □없]음.값을 직접 입력할 수 [□있 □없]음 <div></div>						
콤보상자	<ul style="list-style-type: none">· 목록상자와 비슷, 더 적은 공간 차지. 드롭다운 화살표 클릭시 목록이 표시됨. □□의 값만 선택 가능. <div></div>						
확인란	<ul style="list-style-type: none">· 하나□□을 선택 가능. 각 항목은 예/아니오 선택 <table><tr><td>컨트롤</td><td>예</td><td>아니오</td></tr><tr><td>확인란</td><td><input checked="" type="checkbox"/></td><td><input type="checkbox"/></td></tr></table>	컨트롤	예	아니오	확인란	<input checked="" type="checkbox"/>	<input type="checkbox"/>
컨트롤	예	아니오					
확인란	<input checked="" type="checkbox"/>	<input type="checkbox"/>					
옵션단추	<ul style="list-style-type: none">· 하나□ 선택 가능. 각 항목은 예/아니오 선택. <table><tr><td>컨트롤</td><td>예</td><td>아니오</td></tr><tr><td>옵션 단추</td><td><input checked="" type="radio"/></td><td><input type="radio"/></td></tr></table>	컨트롤	예	아니오	옵션 단추	<input checked="" type="radio"/>	<input type="radio"/>
컨트롤	예	아니오					
옵션 단추	<input checked="" type="radio"/>	<input type="radio"/>					
토글단추	<ul style="list-style-type: none">· 하나□□을 선택 가능. 각 항목은 예/아니오 선택. <table><tr><td>컨트롤</td><td>예</td><td>아니오</td></tr><tr><td>토글 단추</td><td><input type="checkbox"/></td><td><input type="checkbox"/></td></tr></table>	컨트롤	예	아니오	토글 단추	<input type="checkbox"/>	<input type="checkbox"/>
컨트롤	예	아니오					
토글 단추	<input type="checkbox"/>	<input type="checkbox"/>					
레이블	<ul style="list-style-type: none">· 제목, 캡션 등 단순 설명 텍스트를 표시. □□□□컨트롤· 탭 순서에서 □□됨(커서가 들어가지 않음)						
명령 단추	<ul style="list-style-type: none">· 명령 수행을 위한 단추(레코드 인쇄, 찾기 등). □□□□ 컨트롤· 이벤트 프로시저, 매크로 함수 등을 입력하여 □□할 기능 정의						
하위폼/ 보고서	<ul style="list-style-type: none">· 폼 안에 또 다른 □ 또는 □□□를 작성· 1:다 관계 테이블을 표시할 때 주로 사용						

폼 컨트롤 요소별 속성

공통	
IME모드	컨트롤 선택시 □□□□모드 지정('한글' 또는 '영숫자 반자')
탭 인덱스	해당 컨트롤의 □□□□를 지정
자동 고침 설정	사용자가 잘못 입력한 영어단어를 자동으로 □□□□
사용 가능	컨트롤에 □□□□할 수 있는지 여부 설정.
잠금	컨트롤에 입력된 데이터의 □□□□가능 여부 설정. 예=잠금
편집가능	컨트롤의 데이터의 □□□□가능 여부 설정. 예=편집가능
삭제가능	컨트롤의 데이터의 □□□□가능 여부 설정. 예=삭제가능

텍스트 상자	
상태표시줄 텍스트	컨트롤 선택시 □□□□줄 에 표시할 메세지 설정
컨트롤 원본	컨트롤투 바운드(연결)할 □□□□ 지정. 문자는 □□□□표(□□), 필드명은 □□□□(□□) 로 묶어줌
중복내용 숨기기	데이터가 이전 레코드와 □□ 때 컨트롤투 숨김 여부 설정
엔터키 기능	컨트롤투 선택시 □□키를 누를 경우 수행할 기능 설정
누적 합계 속성	누적되며 추가되는 번호의 속성 설정. 모드=모든 데이터 설정. 그룹=그룹 설정. 그룹이 바뀌면 초기화. ※그룹 내의 데이터에 대한 □□ 번호 표시에 사용. 이 경우 컨트롤원본은 '년'로, 누적 합계 속성은 '그룹'으로 설정
콤보 상자	
행 원본	콤보 상에서 사용할 데이터 설정
컨트롤투 원본	콤보 상자에 연결할 데이터 설정
바운드 열	콤보상자에 표시되는 열 중에서 컨트롤 원본 속성에 연결된 필드에 입력할 열을 지정

폼 컨트롤 작성, 수정 및 선택

1. 컨트롤 작성

- **컨트롤투 마법사**를 이용하여 작성 가능
- 컨트롤을 **□□한 순서대로 탭 순서가 지정**됨
- 하나의 폼 안에서는 **같은 이름**을 가진 컨트롤을 만들 수 **[□있 □없]음**
- 사각형, 원 등 **도형 컨트롤**도 추가 **[□가 □불가]능**
- 폼 디자인: **빈 양식**의 폼에서 사용자가 **□□ 컨트롤**들을 추가하는 방식으로 만드는 도구

2. 컨트롤 선택 방법

- **하나**의 컨트롤 선택: **□□**
- **모든** 컨트롤 선택: **Ctrl + □** 또는 마우스로 모든 컨트롤 **드래그**
- **연속적** 컨트롤 선택: 마우스로 원하는 컨트롤 **□□□**
- **비연속적** 컨트롤 선택: **□□□□□ + 마우스 클릭**

3. 탭 순서 지정/변경 방법

- 컨트롤 **직접 작성**시: **작성한 □□대로** 탭 순서가 설정됨
- **마법사/레이아웃 도구**로 폼 생성시 탭 순서: **□→□□, □→□** 방향
- **[□□□] 대화상자**: 컨트롤의 **이름행**을 **드래그**해서 **탭 순서 조정**
- 탭 정지 속성: **탭 정지 속성=□** 인 경우 **포커스가 되지 않도록** 설정 가능(컨트롤투를 제거하는 것은 아님)

4. 컨트롤 복사/이동

- **다른 구역**에서 붙여넣기: **붙여넣기 구역의 □□ □**에 붙여짐
- **같은 구역**에서 붙여넣기: **복사한 컨트롤의 바로 □□**에 붙여짐
- 이동: **□□□□□키**를 누르고 **이동시 가로/세로맞춤 □□□□+방향키** → 컨트롤 위치를 **세밀하게 변경**

5. 컨트롤 크기 변경

- 크기 변경: **□□□□□ + 방향키** → 컨트롤 **크기**를 **세밀하게 변경**
- 그룹화
 - **그룹화하기**: 여러 컨트롤 선택 → [폼 디자인 도구] > [정렬] > [크기 및 순서 조정] > [크기/공간] > **[□□] 선택**
 - **그룹 해제**: ‘그룹화하기’에서 [그룹] 대신 **[□□□□] 선택**

조건부 서식

- 원하는 필드값에 대한 서식 지정
- ‘**□□□**’, ‘**□**’, ‘**필드에 □□□가 있음**’ 세 가지 기준 중 택1 가능
- 식이 TRUE/FALSE로 평가되는 경우에 대한 서식 지정
- **최대 □□개**의 규칙 설정 가능(엑세스 2010 기준)
- 우선순위 변경 가능. **□에 오는 규칙을 우선 적용**
- 기본 서식 적용 가능: **모든 조건부 서식의 조건에 일치하지 □□ 값**에 적용됨
- ‘다른 레코드와 비교’ 규칙 유형 선택시 비교한 차이를 막대그래프로 표시

하위 폼

1. 하위 폼 개념

- **하위폼** = 기본 폼 안에 **□□된 폼**
- **일대□** 관계 테이블 데이터 표시에 유용

- **일=□폼 / 다=□폼**
- 기본폼-하위폼의 **연결 필드**는 **데이터형식, 필드 크기**가 **□□야 함**(=호환)
- **하위 폼 갯수 제한** **[□0 □X]**, 중첩 수준은 **최대 □수준**까지 가능
- 기본 폼-하위 폼 필드는 ‘**□□□□□□**’ 속성으로 지정 가능([하위 폼 필드 연결기]로 속성 지정)
- **하위 폼 □□□**: 기본폼-하위폼의 연결 필드 지정. 기본폼, 하위폼 연결 필드를 동시에 지정 가능. 두 개 이상의 필드도 지정 가능.

2. 하위 폼에서 가능한 것들

- 테이블 간 **관계 설정 안되어있어도 하위폼으로 연결 [□가 □불가]능**
- 여러 개의 연결 필드 지정 가능. 구분자는 **□□□□(□)**

3. 하위 폼에서 불가능한 것들

- 하위 폼은 **연속 폼**의 형태로 표시 가능(**기본 폼은 [□가 □불가]능**)

분할 표시 폼

- **폼 보기**와 **데이터 시트 보기**를 **□□□에 동시에** 보여줌
- 폼과 데이터 시트는 **같은 □□□□□□에 연결, 상호 □□화**됨
- **한쪽에서 필드**를 선택하면 **다른 쪽에도 □□한 필드가 선택**됨
- [만들기]-[폼]-[폼 분할] 클릭하여 생성
- 컨트롤의 크기 조정 가능, 데이터시트 표시 위치 설정 가능

보고서 활용

보고서의 개념과 특징

- **보고서**: 테이블/쿼리의 내용을 **화면/프린터로 □□**하기 위한 개체
- 필수 요소: ①**레코드 원본**: **□□□, □□, □□□문** ②**□□□□**(보고서에 데이터 표시)
- 가능한 기능:
 - ①**하나 이상의 □□ 또는 □□**에서 **필드 선택** 가능
 - ②**데이터 □□화/□□도** 가능 ③**조건부 □□** 적용 가능
 - ④**□□글/□□글**(머리글은 첫페이지, 바닥글은 마지막 페이지에 1번 표시)
- 불가능한 기능: ①보고서의 컨트롤로 **원본 데이터 수정 [□가 □불가]능**

보고서의 영역

보고서 머리글	보고서의 맨 □□□□ 위에 한 번만 표시 로고, 보고서 제목, 날짜 등이 표시
페이지 머리글	□□□□의 맨 위에 출력
그룹 머리글	각 레코드 □□의 맨 앞에 표시 그룹별 이름, 요약 정보 등을 삽입
본문	레코드 원본의 모든 □에 대해 □ 번씩 출력 보고서의 본문을 구성하는 컨트롤이 여기에 추가됨
그룹 바닥글	각 레코드 □□의 맨 끝에 표시 그룹별 이름, 요약 정보 등을 삽입
페이지 바닥글	□□□□의 맨 밑에 출력
보고서 바닥글	보고서의 맨 □□□□□□ 아래에 한 번 출력 로고, 보고서 제목, 날짜 등이 표시

- 각 머리/바닥글은 **합수를 이용해 감싸는 영역**의 **□□정보**를 표시 가능
예: 페이지 머리글 = 페이지 내 레코드의 총합 표시 가능

보고서의 속성

레코드 원본	한 개 이상의 테이블 또는 쿼리에서 □□를 선택 하여 레코드 원본으로 지정 [속성 시트]>[레코드 원본]>테이블/쿼리를 선택 보고서 마법사를 통해 필드를 선택하여 지정
필터	레코드 □□ 추출 조건 설정
필터 사용	필터 사용 여부 지정
정렬 기준	정렬 기준 지정
반복 실행 구역	□□□□을 각 페이지에 모두 표시할 지 여부
기본 보기	인쇄 미리보기: □□에 출력되는 그대로 페이지를 구분하여 표시 보고서 보기: 페이지 구분 □이 □장으로 표시

보고서의 정렬과 그룹화

- **그룹화**: 특정 ☐의 값을 기준으로 데이터를 그룹화하는 기능
- 그룹화 기준 필드를 선택하면 **필드의 값이 ☐은 레코드를 ☐차순으로 정렬**하여 표시(정렬기준은 사용자가 수정 가능)
- 그룹 머리글과 그룹 바닥글에 **그룹별 ☐ 정보를 삽입**
 - 여러 필드에 요약 추가, 같은 필드에 여러 종류 요약 계산 가능
 - 그룹화를 하려면 ☐글 또는 ☐글 중 최소한 하나는 설정해야 함
 - 머리글과 바닥글을 제거시 **그룹이 제거**☐됨 ☐되진 않음
 - **그룹을 제거하면 머리글과 바닥글은 제거**☐됨 ☐되진 않음
- 필드를 기준으로 **최대 ☐단계까지 그룹화**가 가능, 같은 필드나 식도 **두 번 이상 그룹화** ☐가 ☐불가]능
- 정렬 기준 데이터 유형
 - 날짜: 전체 값(기본), 일, 주, 월, 분기, 연도 중 선택한 기준으로 그룹화
 - 문자: 전체 필드/처음 첫-다섯자까지를 기준으로 그룹화

조건부 서식

- 사용자가 설정한 조건에 부합할 경우 해당 조건에 설정된 서식으로 표시
- 부합하지 않는 경우에 대해 **따로 조건부 서식 설정은 ☐가 ☐불가]능**
- ☐ 서식: 설정한 모든 조건에 **모두 부합하지 않을 경우 표시할 서식**
- 한 컨트롤에 **최대 ☐개까지**의 조건부 서식 설정 가능할 수 있습니다
- **필드 ☐, ☐ 필드에 ☐가 있음** 3가지 규칙 중 하나를 설정
 - 필드에 ☐: 필드에 ☐가 있는지 여부에 따라 서식 지정
- **다른 ☐ ☐으로 출력하거나 내보내면 조건부 서식은 해제**됨
- 여러 조건부 서식의 조건이 **충돌**할 경우, ☐ 번째 조건부 서식이 먼저 적용
- 조건을 지정할 때 **와일드 카드 문자(*, ?)는 사용 할 수 ☐있 ☐없]음**

페이지 번호

- [Page]: 현재 페이지 표시
- [Pages]: 전체 페이지 수 표시
- **문자열은 ☐표(☐)로 감싸**줌
- 문자열을 **연결**할 때는 ☐ 연산자를 사용
- Format(“값”, “형식”): 이용해 “값”을 지정한 “형식”으로 표시

보고서의 종류

1. 하위보고서

- **일대다 관계**가 적용되어있는 테이블/쿼리의 데이터 표시에 유용
- 일대다에서 **일=☐보고서, 다=☐보고서**
- 레코드 원본 사이에 ☐를 **만들어야 함**
- **하위 보고서 ☐을 이용해 작성 가능**
- ☐으로 **연결**: 테이블/쿼리가 서로 일대다로 설정 + 주 보고서에 기본키가 있고 하위보고서에 기본키 필드와 같거나 호환되는 필드가 있는 경우
- 가능:
 - ① 디자인 보기 화면에서 **하위보고서 ☐조절, ☐ 가능.**
 - ② 디자인 보기 화면에서 **하위보고서 컨트롤의 레이블 ☐ 가능**
 - ③ ☐화, ☐ 기능 설정 가능(주보고서, 하위보고서 모두)
 - ④ ☐개 수준까지 하위 보고서 **중첩 가능**

2. 우편 레이블 보고서

- **우편물 레이블(우편 봉투)**를 만들어주는 보고서
- **우편물 레이블 마법사**로 간편하게 만들 수 있음
 - 레이블 크기, 텍스트 모양, 사용 가능한 필드, 정렬 기준 등을 지정
 - 마지막 단계: ‘인쇄될 우편물 레이블 미리 보기’ 선택→ 완성된 보고서가 ☐ 상태로 표시됨
- 사용자가 크기, 형식 등을 지정하는 레이블도 사용 가능
- ☐ 용지/☐ 용지 사용
- **필드명을 입력해 필드명에 해당하는 값을 활용**
 - 추가 가능한 필드의 갯수는 **최대 ☐개**
 - ☐필드명☐문자열 형식으로 작성 - 필드 값 뒤에 문자열을 붙여 출력

3. 기타 보고서

- ☐ 보고서: 테이블/쿼리를 원본으로 한 차트 작성
- ☐ 보고서: 데이터를 **세금계산서, 거래명세서** 등 기업에서 주로 사용하는 양식으로 출력하기 위한 보고서

보고서 도메인 계산 함수

1. 정의

- 보고서의 **레코드셋**에 대한 ☐를 계산하기 위한 함수
- 형식: **=DLOOKUP(“[필드]”, “[도메인(테이블/쿼리)]”, “조건”)**

2. 종류

DLOOKUP	조건에 해당하는 레코드의 특정 필드 <input type="checkbox"/>
DCOUNT	조건에 해당하는 레코드의 <input type="checkbox"/>
DSUM	조건에 해당하는 레코드의 <input type="checkbox"/>
DAVG	조건에 해당하는 레코드의 <input type="checkbox"/>
DMAX	조건에 해당하는 레코드의 <input type="checkbox"/> 값(가장 <input type="checkbox"/> 값)
DMIN	조건에 해당하는 레코드의 <input type="checkbox"/> 값(가장 <input type="checkbox"/> 값)

3. 예시

- **=DSUM(“매출”, “판매실적”, “지점=지점입력”)**
 - “판매실적” 테이블에서 “지점”이 “지점입력” 컨트롤의 값과 같은 레코드들의 “매출”의 총 합

보고서 인쇄

1. 보고서 인쇄 미리보기

- ☐에 **출력되는 그대로** 페이지를 구분하여 표시하는 화면
- ☐ 대화상자를 통해 인쇄 설정 가능

2. 보고서 인쇄 페이지 설정 대화상자

- **인쇄옵션, 페이지, 열** 탭으로 구성
- **인쇄옵션**: ☐, ☐만 인쇄, 분할표시 폼
 - ☐: 위쪽, 아래쪽, 왼쪽, 오른쪽 여백을 밀리미터 단위로 설정
 - ☐만 인쇄: 그래픽 요소(테두리, 눈금선 등)는 생략하고 데이터만 인쇄
- **페이지**: 용지☐, 용지 ☐, 프린터 선택
- **열**: ☐ 설정, ☐ 크기, 열 레이아웃
 - ☐ 크기 ‘본문과 같게’: 열의 너비(너비(I) 옆의 숫자상자)와 높이(높이(E) 옆의 숫자상자)를 보고서 본문의 너비와 높이에 맞춘 값으로 수정
 - 열 크기에 비해 **페이지의 가로 크기가 작은 경우 ☐서 인쇄**됨

|| 매크로 활용

매크로의 개념

1. 매크로와 구성 요소

- **매크로**: ☐ 작업을 자동으로 수행하기 위한 기능
- 프로시저, 형식, 데이터 선언과 정의 등의 선언 집단
- **매크로 ☐ ☐ ☐ ☐**
- ☐: 명령문의 집합
- ☐: 프로시저의 집합

2. 매크로로 가능한 것들

- ① 폼, 보고서, 테이블, 쿼리 등에 사용 가능
- ② **복사&붙여넣기**: 이미 매크로에 추가한 동작 반복 ☐가 ☐불가]능
- ③ **여러 매크로 함수 포함 가능. 하위 매크로 포함 ☐가 ☐불가]능**
- ④ 매크로를 컨트롤의 ☐ 속성에 포함
- ⑤ **한 단계씩 실행 ☐가 ☐불가]능(Ctrl + ☐ 키)**
- ⑥ 폼을 열고 닫거나 메시지 박스 표시 등 다양한 기능 수행 가능

3. 자동실행 매크로

- **데이터베이스 ☐을 열 때** 자동으로 실행되는 매크로
- 매크로 이름을 ‘☐’로 설정하면 됨
- **자동실행 매크로 실행 방지**: ☐키를 누른 채 데이터베이스 파일을 엮

프로시저와 변수

1. 프로시저의 정의와 종류

- **프로시저**: 명령문의 집합. 모듈을 이루는 구성요소

SUB ~ END SUB	결과값 반환 <input type="checkbox"/> <input type="checkbox"/> X
FUNCTION ~ END FUNCTION	결과값 반환 <input type="checkbox"/> <input type="checkbox"/> X
PROPERTY ~ END PROPERTY	결과값 반환 <input type="checkbox"/> <input type="checkbox"/> X. 개체의 속성을 새로 정의할 때 사용

2. 변수

- **변수**: 컴퓨터가 명령을 처리하는 도중 발생하는 값을 저장하는 값. **□□적** (명령에 따라 □□ 가능)
- 변수의 주요 자료형: ① VARIANT(데이터 형식 생략시) ② INTEGER(정수) ③ STRING(문자열)
- 변수의 SCOPE:

PUBLIC	□□의 처음에 선언. 선언한 모듈 안의 모든 프로시저 & 다른 모든 모듈에서 사용 [□가 □불가]능
PRIVATE	□□의 처음에 선언. 선언한 모듈 안의 모든 프로시저 에 사용 가능. 다른 모듈에서 사용 [□가 □불가]능
STATIC	□□□□의 처음에 선언. 선언한 프로시저 에서 사용 가능. 프로시저 종료 후에도 값 유지
DIM	□□□□의 처음에 선언한 경우: 선언한 프로시저 안에서만 사용 가능 □□의 처음에 선언한 경우: 선언한 모듈의 모든 프로시저 에서 사용 가능. 다른 모듈에서 사용 [□가 □불가]능

모듈

1. 모듈의 정의와 종류

- **모듈**: □□□□의 **집합**. 표준 모듈 vs 클래스 모듈로 나뉨.
- 모듈의 종류

□□모듈	① 워크시트 모듈: Sheet로 표시 ② ThisWorkbook 모듈 ③ 공용 모듈: 일반적으로 사용하는 모듈
□□□ 모듈	① 폼 모듈: 특정 폼과 연결 ② 보고서 모듈: 특정 보고서와 연결

매크로 함수 - 실행, 가져오기/내보내기

1. 실행

□□□□Access	액세스 종료	□□□Application	응용 프로그램 실행
□□□□Window	창 닫기	Run□□□	SQL문 실행
Open□□□□□	보고서 열기	Run□□□□□	매크로 실행
Open□□□□	폼 열기	Run□□□□	프로그램 실행
Open□□□□□	쿼리 열기	MessageBox	메시지 상자 표시
Open□□□□□	테이블 열기	CancelEvent	이벤트 취소

2. 검색, 포커스

GoTo□□□□□□	컨트롤로 포커스 이동
GoTo□□□□	페이지로 포커스 이동
GoTo□□□□□□	레코드로 포커스 이동
□□□□Record	조건을 만족하는 첫 번째 레코드로 이동
Find□□□□Record	조건을 만족하는 레코드 중 현재 레코드 바로 다음 레코드로 이동

3. 가져오기/내보내기

□□□□□□ExportData	데이터베이스 파일과 내보내기/가져오기/연결 지원
□□□□□□Export Spreadsheet	스프레드시트 파일과 내보내기/가져오기/연결 지원
□□□□□□ExportText	텍스트 파일과 내보내기/가져오기/연결 지원
□□□□□□With Formatting	데이터베이스 개체를 형식을 지정하여 내보내기 - 테이블/쿼리: 엑셀(xls/xlsx), 텍스트(txt/rft), html, pdf, xps - 보고서: 엑셀(xls/xlsx), 텍스트(txt/rft), html, pdf xps + snp

Form 개체 함수

Record□□□□□□	데이터 원본 지정
□□□□□□ble	폼의 보여지기 여부 설정
Re□□□□□□	데이터 원본을 다시 갱신 후 업데이트
Set□□□□□□	포커스를 줌

DoCmd 개체 함수

Open□□□□□□	보고서 호출	Run□□□□	SQL문 실행
Open□□□□□	폼 호출	Run□□□□□□	매크로 실행
Open□□□□□□	쿼리 호출	□□□□Command	액세스 내부 제공 명령 실행
□□□□□	활성 개체 닫기		
□□□□	액세스 종료	Re□□□□□□	개체 데이터 수정 및 결과 반영

Recordset 개체 함수

1. 속성

□□□□□□Type	커서 유형
□□□□□□Type	잠금 유형
Record□□□□□□	레코드의 갯수
□□□□□	정렬 기준
□□□□□□Connection	연결되어 있는 Connection 개체
□□□□□□OF	현재 레코드의 위치가 레코드셋의 앞에 오는지 여부
□□□□□□OF	현재 레코드의 위치가 레코드셋의 뒤에 오는지 여부

2. 함수

□□□□□	레코드셋 열기	□□□□□□	레코드 값 변경 적용
□□□□□□	레코드셋 닫기		
□□□□□□New	새 레코드 추가	□□□□□	기준에 부합하는 레코드 찾기
□□□□□□□	레코드 삭제		
□□□□□	인덱스가 지정된 레코드 중 기준에 부합하는 레코드를 찾고 커서를 이동시킴		

이벤트 프로시저

- **특정 이벤트가 발동하는 경우 자동으로 실행**되는 프로시저
- 특정 이벤트 발동 시 시행하고자 하는 **동작을 이벤트 프로시저 안에 정의**

폼 관련	
□□□□	폼을 열려고 할 때
□□□□	폼이 열린 후에
□□□□□□ialize	폼 초기값이 설정되었을 때
Re□□□□□	폼의 크기가 변경되었을 때
Un□□□□□	폼을 닫으려고 할 때
□□□□□	폼이 닫혔을 때
포커스 관련	
□□□□□ivate	폼/보고서가 활성화되었을 때
□□□□□□ivate	폼/보고서가 비활성화 되었을 때(다른 폼/보고서가 활성화되었을 때)
□□□□□Focus	컨트롤/폼이 포커스를 얻었을 때
□□□□□□Focus	컨트롤/폼이 포커스를 잃었을 때(다른 컨트롤/폼으로 포커스가 이동)
키보드/마우스 관련	
□□□□□	마우스 클릭시
□□□□□Click	마우스 더블클릭시
□□□□□□	컨트롤의 텍스트 변경시
After□□□□□□□	새 레코드 추가 시
□□□□□□□Insert	새 레코드가 생기기 전
□□□□□□□Update	레코드 값 변경 후
Before□□□□□□□	레코드 값 변경 전