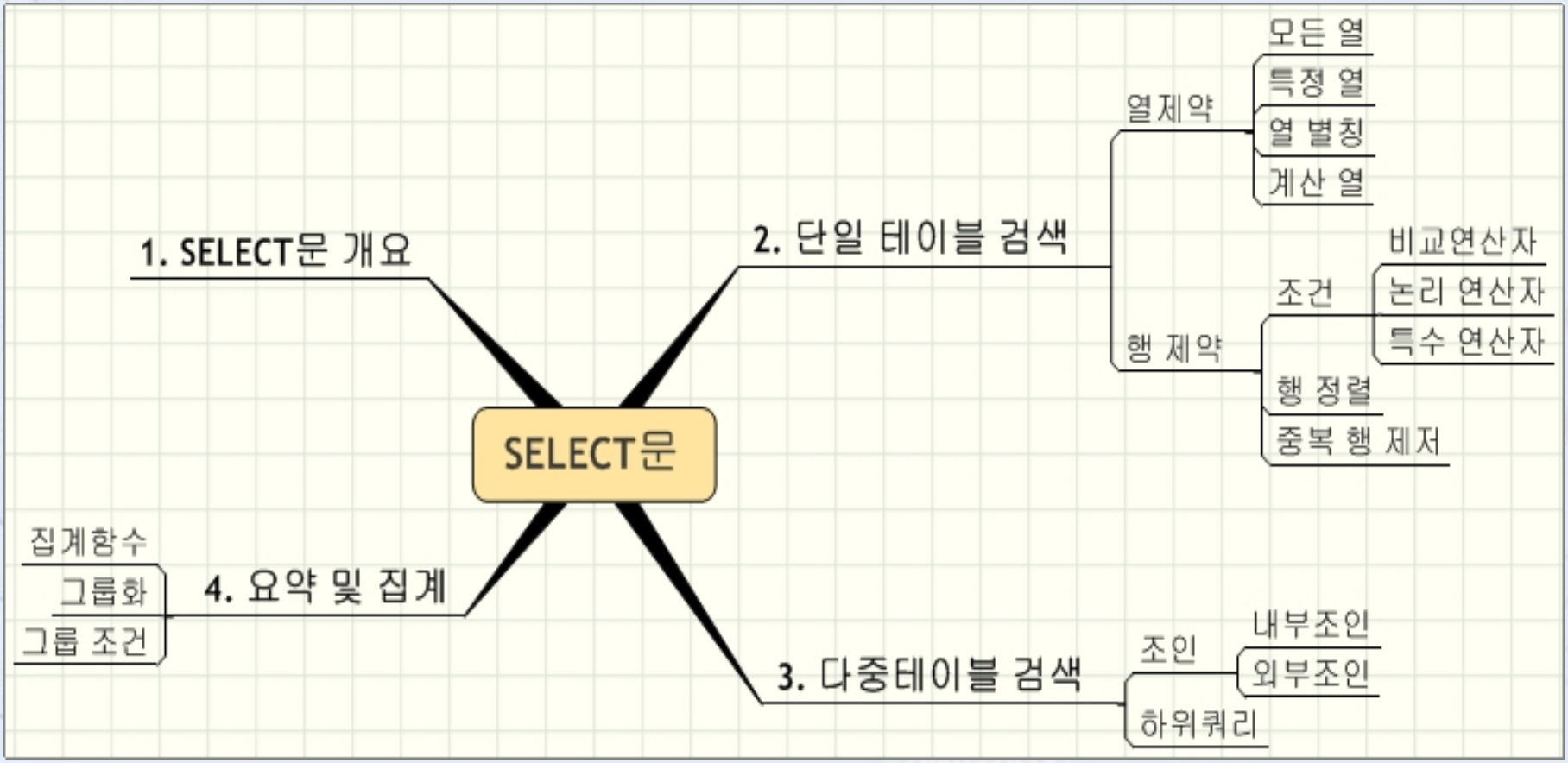
제5장 SQL


SELECT문
1. SELECT문 개요
2. 단일 테이블 검색
열제약
모든 열
특정 열
열 별칭
계산 열
행 제약
조건
비교연산자
논리 연산자
특수 연산자
행 정렬
중복 행 제저
3. 다중테이블 검색
조인
내부조인
외부조인
하위쿼리
4. 요약 및 집계
집계함수
그룹화
그룹 조건

1. SELECT 문 개요

- **BNF(Backus Normal Form)**
  - **BNF는 프로그래밍 언어의 구문을 정의하는 데 많이 사용되는 표기법**

[표 2-1] 중요한 BNF 구성 요소들

구성 요소	설명
::=	왼쪽의 항목은 오른쪽과 같이 정의된다.
〈element〉	다른 곳에 정의되어 있는 항목
[ ]	생략 가능하다.
{ ... | ... }	여러 항목 중에서 하나를 선택할 수 있다.
...n	여러 번 반복될 수 있다.
name, expression	이름, 연산식 등으로 대치시킬 곳

1. SELECT 문 개요

- **기본적인 SELECT 문의 구문**

```
SELECT        [DISTINCT ]  속성
FROM          테이블 [ ,...n ]
[ WHERE       검색조건 ]
[ GROUP BY    속성 ]
[ HAVING      검색조건 ]
[ORDER BY     속성 [ASC|DESC] ];
```

SQL문의 작성 방법

- SQL 문장은 대소문자를 구별하지 않는다.
- SQL 문장은 한 줄 또는 여러 줄에 입력될 수 있다.
- 일반적으로 키워드는 대문자로 입력한다. 다른 모든 단어, 즉 테이블 이름, 열 이름은 소문자로 입력한다.(권장)
- SQL문 마지막 절의 끝에 ";"를 기술하여 명령의 끝을 표시 한다.

2. 열 선택 방법

SELECT [ALL | DISTINCT] < select_list >

지정 방법	예시
모든 열 지정	**SELECT ***
특정 열 지정	**SELECT name, sex, address, phone**
중복 제거	**SELECT DISTINCT name**
열 별칭 지정	**SELECT name AS 성명, sex, address as 주소**
계산열 지정	**SELECT name, (본봉 * 0.1) AS 세금**

2. 열 선택 방법

1. 모든 열(*) 지정

제품							
제품번호	제품이름	공급업체번호	제품분류번호	포장단위	단가	재고량	발주량
1	태양 100% 오렌지 주스	서울 무역 ㈜	1	10 boxes x 20 bags	₩80,000	39	0
2	태양 100% 레몬 주스	태양 식품 ㈜	1	24 - 12 oz bottles	₩19,000	17	40
3	태양 체리 시럽	태양 식품 ㈜	2	12 - 550 ml bottles	₩10,000	13	70
4	신한 100% 복숭아 시럽	신한 식품 ㈜	2	48 - 6 oz jars	₩22,000	53	0
5	신한 100% 파인애플 시럽	신한 식품 ㈜	2	36 boxes	₩21,000	0	0

예제: 제품 테이블의 모든 열을 검색

```
SELECT  *
FROM    제품 ;
```

제품번호	제품이름	공급업체번호	제품분류번호	포장단위	단가	재고량	발주량
1	태양 100% 오렌지 주스	서울 무역 ㈜	1	10 boxes x 20 bags	₩80,000	39	0
2	태양 100% 레몬 주스	태양 식품 ㈜	1	24 - 12 oz bottles	₩19,000	17	40
3	태양 체리 시럽	태양 식품 ㈜	2	12 - 550 ml bottles	₩10,000	13	70
4	신한 100% 복숭아 시럽	신한 식품 ㈜	2	48 - 6 oz jars	₩22,000	53	0
5	신한 100% 파인애플 시럽	신한 식품 ㈜	2	36 boxes	₩21,000	0	0

2. 열 선택 방법

2. 특정 열 지정

제품							
제품번호	제품이름	공급업체번호	제품분류번호	포장단위	단가	재고량	발주량
1	태양 100% 오렌지 주스	서울 무역 ㈜	1	10 boxes x 20 bags	₩80,000	39	0
2	태양 100% 레몬 주스	태양 식품 ㈜	1	24 - 12 oz bottles	₩19,000	17	40
3	태양 체리 시럽	태양 식품 ㈜	2	12 - 550 ml bottles	₩10,000	13	70
4	신한 100% 복숭아 시럽	신한 식품 ㈜	2	48 - 6 oz jars	₩22,000	53	0
5	신한 100% 파인애플 시럽	신한 식품 ㈜	2	36 boxes	₩21,000	0	0

예제: 제품 테이블에서 제품이름, 단가, 포장단위를 검색

SELECT 제품이름, 단가, 포장단위

FROM 제품 ;

제품이름	단가	포장단위
태양 100% 오렌지 주스	₩80,000	10 boxes x 20 bags
태양 100% 레몬 주스	₩19,000	24 - 12 oz bottles
태양 체리 시럽	₩10,000	12 - 550 ml bottles
신한 100% 복숭아 시럽	₩22,000	48 - 6 oz jars
신한 100% 파인애플 시럽	₩21,000	36 boxes

2. 열 선택 방법

3. 중복 제거

제품							
제품번호	제품이름	공급업체명	제품분류번호	포장단위	단가	재고량	발주량
1	태양 100% 오렌지 주스	서울 무역 ㈜	1	10 boxes x 20 bags	₩80,000	39	0
2	태양 100% 레몬 주스	태양 식품 ㈜	1	24 - 12 oz bottles	₩19,000	17	40
3	태양 체리 시럽	태양 식품 ㈜	2	12 - 550 ml bottles	₩10,000	13	70
4	신한 100% 복숭아 시럽	신한 식품 ㈜	2	48 - 6 oz jars	₩22,000	53	0
5	신한 100% 파인애플 시럽	신한 식품 ㈜	2	36 boxes	₩21,000	0	0

예제: 제품 테이블에서 공급업체명을 중복을 제거하고 검색

```
SELECT  DISTINCT 공급업체명
FROM    제품 ;
```

공급업체
서울 무역 ㈜
태양 식품 ㈜
신한 식품 ㈜

```
SELECT  공급업체명
FROM    제품 ;
```

공급업체명
서울 무역 ㈜
태양 식품 ㈜
태양 식품 ㈜
신한 식품 ㈜
신한 식품 ㈜

2. 열 선택 방법

4. 열 별칭 지정

제품							
제품번호	제품이름	공급업체명	제품분류번호	포장단위	단가	재고량	발주량
1	태양 100% 오렌지 주스	서울 무역 ㈜	1	10 boxes x 20 bags	₩80,000	39	0
2	태양 100% 레몬 주스	태양 식품 ㈜	1	24 - 12 oz bottles	₩19,000	17	40
3	태양 체리 시럽	태양 식품 ㈜	2	12 - 550 ml bottles	₩10,000	13	70
4	신한 100% 복숭아 시럽	신한 식품 ㈜	2	48 - 6 oz jars	₩22,000	53	0
5	신한 100% 파인애플 시럽	신한 식품 ㈜	2	36 boxes	₩21,000	0	0

예제: 제품 테이블에서 제품이름, 단가 재고량 검색, 단 제품이름을 제품명으로 표시

SELECT 제품이름 AS 제품명, 단가, 재고량

FROM 제품 ;

제품명	단가	재고량
태양 100% 오렌지 주스	₩80,000	39
태양 100% 레몬 주스	₩19,000	17
태양 체리 시럽	₩10,000	13
신한 100% 복숭아 시럽	₩22,000	53
신한 100% 파인애플 시럽	₩21,000	0

2. 열 선택 방법

5. 계산열 지정

제품							
제품번호	제품이름	공급업체명	제품분류번호	포장단위	단가	재고량	발주량
1	태양 100% 오렌지 주스	서울 무역 ㈜	1	10 boxes x 20 bags	₩80,000	39	0
2	태양 100% 레몬 주스	태양 식품 ㈜	1	24 - 12 oz bottles	₩19,000	17	40
3	태양 체리 시럽	태양 식품 ㈜	2	12 - 550 ml bottles	₩10,000	13	70
4	신한 100% 복숭아 시럽	신한 식품 ㈜	2	48 - 6 oz jars	₩22,000	53	0
5	신한 100% 파인애플 시럽	신한 식품 ㈜	2	36 boxes	₩21,000	0	0

예제: 제품 테이블에서 제품이름, 단가, 재고량, 재고액을 검색.
재고액은 단가에 수량을 곱해서 계산한다.

```
SELECT  제품이름, 단가, 재고량, [단가]*[재고량] AS 재고액
FROM    제품 ;
```

제품이름	단가	재고량	재고액
태양 100% 오렌지 주스	₩80,000	39	₩3,120,000
태양 100% 레몬 주스	₩19,000	17	₩323,000
태양 체리 시럽	₩10,000	13	₩130,000
신한 100% 복숭아 시럽	₩22,000	53	₩1,166,000
신한 100% 파인애플 시럽	₩21,000	0	₩0

3. 행 선택 방법

SELECT < select_list >

FROM { <table_source> } [,...n]]

[WHERE <search_condition>]

[<GROUP BY>]

[HAVING < search_condition >]

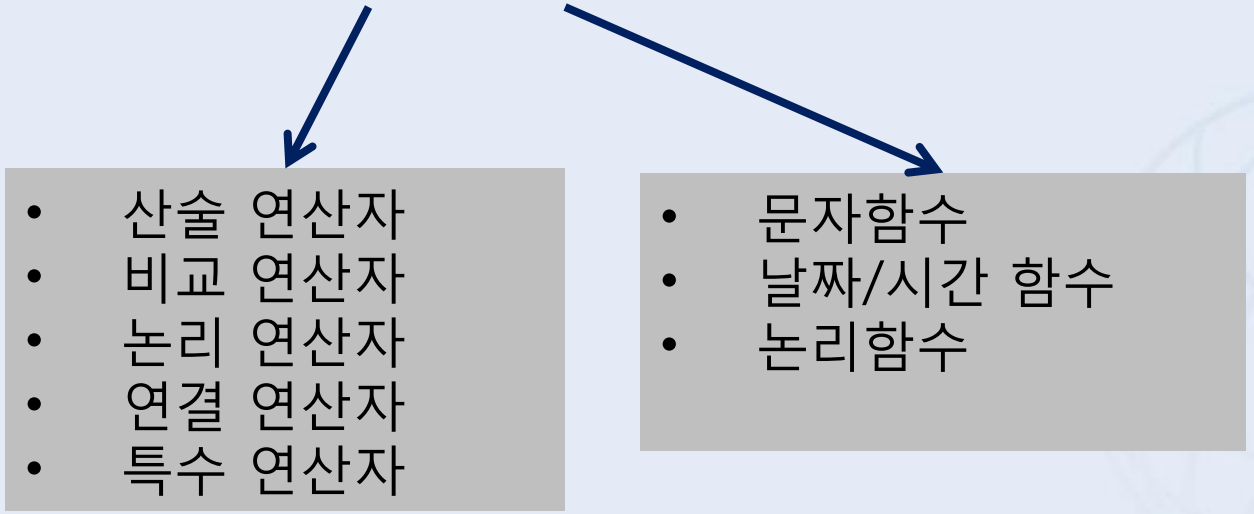
< search_condition > : 열이름 연산자 값

반환될 행이 충족해야 할 조건을 정의합니다.

검색 조건에 포함시킬 수 있는 조건자의 개수에는 제한이 없습니다.

3. 행 선택 방법

SELECT *열이름*
FROM *테이블명*
WHERE *열이름 연산자 값*;



- 산술 연산자
- 비교 연산자
- 논리 연산자
- 연결 연산자
- 특수 연산자

- 문자함수
- 날짜/시간 함수
- 논리함수

2014년 4월 ITQ Access 쿼리 문제

2. [테이블1:부품재고관리]를 이용하여 다음과 같은 조건에 따라 쿼리를 완성하시오. (90점)

≪조건≫

(1) 쿼리 이름 : 부품재고관리현황

(2) 부품구분 : 관리코드의 첫 번째 글자가 'R'이면 '로봇', 'S'이면 '센서', 'V'이면 '진공', 'P'이면 '파워'로 적용(SWITCH, LEFT 함수 사용)

(3) 연간보유량 : '월소모량 × 12'의 값에 여유배율을 곱한 값으로 계산하되 최대값은 '70'으로 적용. 단, 여유배율은 보관소코드의 첫 글자로 적용(IIF, LEFT 함수 사용)

(4) 다음입고일 : 입고일에 90일을 더한 값으로 적용하되, 다음입고일이 일요일이면 다음날인 월요일로 적용(IIF, DATEADD, WEEKDAY 함수 사용)

(5) 입고일에 대해 내림차순으로 정렬

≪출력형태≫

관리코드	부품구분	부품명	입고일	재고량	연간보유량	다음입고일	월소모량	보관소코드
R0004	로봇	1축 로봇	2014-03-30	2	70	2014-06-28	10	10004
V0009	진공	저진공 장치	2014-03-10	5	70	2014-06-09	13	20002
S0008	센서	거리센서	2014-02-20	10	70	2014-05-21	7	20001
S0006	센서	열센서	2014-02-05	20	60	2014-05-06	5	10006
P0001	파워	DC 파워	2014-02-01	5	70	2014-05-02	15	10001
R0005	로봇	2축 로봇	2014-01-19	5	48	2014-04-19	4	10005
S0007	센서	광센서	2014-01-10	9	70	2014-04-10	18	10007
P0002	파워	RF 파워	2014-01-06	30	60	2014-04-07	5	10002
P0003	파워	특수 파워	2014-01-05	4	70	2014-04-05	14	10003
V0010	진공	고진공 장치	2014-01-01	10	70	2014-04-01	5	20003

2014년 4월 ITQ Access 쿼리 문제

3. [테이블1:부품재고관리]와 [테이블2:보관소정보]를 이용하여 다음과 같은 조건에 따라 쿼리를 완성하시오. (80점)

≪조건≫

(1) 쿼리 이름 : 부품재고관리현황 분석

(2) 테이블조인 : '보관소코드'를 기준으로 관계 설정(조건 : 두 테이블의 조인된 필드가 일치하는 행만 포함)

(3) 한국을 제외한 지역의 보관소에서 재고량이 '10' 이상인 데이터를 추출하고, 지역을 기준으로 정렬하여 ≪출력형태≫와 같이 표시되는 선택 쿼리를 작성하시오.

≪출력형태≫

보관소코드	지역	등록일	월보관료	부품명
20003	미국	2012-05-01	₩2,000,000	고진공 장치
20001	베트남	2013-04-01	₩700,000	거리센서

3. 행 선택 방법

연산자

산술 연산자

연산자	의미	예제
+	두 숫자를 더한다.	[소계]+[판매세]
-	두 숫자의 차를 구한다.	[가격]-[할인 금액]
*	두 숫자를 곱한다.	[수량]*[가격]
/	두 번째 숫자로 첫 번째 숫자를 나눈다.	[합계]/[항목 수]
%	나누기의 나머지를 구한다.	[투숙객 수] % [객실 수]

3. 행 선택 방법

연산자

비교 연산자

연산자	의미	예제
<	첫 번째 값이 두 번째 값보다 작은 경우 True를 반환한다.	값1 < 값2
<=	첫 번째 값이 두 번째 값보다 작거나 같은 경우 True를 반환한다.	값1 <= 값2
>	첫 번째 값이 두 번째 값보다 큰 경우 True를 반환한다..	값1 > 값2
>=	첫 번째 값이 두 번째 값보다 크거나 같은 경우 True를 반환한다.	값1 >= 값2
=	첫 번째 값이 두 번째 값과 같은 경우 True를 반환한다.	값1 = 값2
<>	첫 번째 값이 두 번째 값과 다른 경우 True를 반환한다.	값1 <> 값

3. 행 선택 방법

연산자

논리 연산자

연산자	용도	예제
And	식1과 식2가 참인 경우 True를 반환한다.	식1 And 식2
Or	식1 또는 식2가 참인 경우 True를 반환한다.	식1 Or 식2
Eqv	식1과 식2가 둘 다 참이거나 둘 다 거짓인 경우 True를 반환한다.	식1 Eqv 식2
Not	식이 참이 아닌 경우 True를 반환한다.	Not 식
Xor	식1과 식2 중에서 하나만 참인 경우 True를 반환한다.	식1 Xor 식2

3. 행 선택 방법

연산자

연결 연산자

연산자	의미	예제
&	문자열 두 개를 결합하여 하나의 문자열을 구성합니다.	문자열1 & 문자열2
+	문자열 두 개를 결합하여 하나의 문자열을 구성하고 Null 값을 전파합니다. 값 하나가 Null이면 전체 식이 Null이 됩니다.	문자열1 + 문자열2

3. 행 선택 방법

연산자

특수 연산자

연산자	의미	예제
Is Null 또는 Is Not Null	값이 Null인지 Not Null인지를 결정합니다.	필드1 Is Not Null
Like "패턴"	와일드카드 문자를 사용하여 문자열 값을 일치시킵니다.	필드1 Like "*백화점"
Between 값1 And 값2	필드의 값이 지정된 값 범위 내에 있는지 확인합니다	필드1 Between 1 And 10 - 또는 - 필드1 Between #07-01-07# And #12-31-07#
In(값1,값2...)	필드의 값이 목록 내의 값과 일치하는 값이 있는지 확인합니다.	필드1 In ("빨강","녹색","파랑") - 또는 - 필드1 In (1,5,7,9)

3. 행 선택 방법

연산자

ANSI-89 와일드카드 문자

문자	설명	예제
*	임의의 수의 문자와 같습니다. 문자열 내의 원하는 위치에 별표(*)를 사용할 수 있습니다.	wh*는 what, white 및 why를 찾으며 awhile 또는 watch는 찾지 않습니다.
?	임의의 알파벳 문자 한 개와 같습니다.	B?ll은 ball, bell 및 bill을 찾습니다.
[ ]	대괄호로 묶인 임의의 문자 한 개와 같습니다.	B[ae]ll은 ball과 bell을 찾으며 bill은 찾지 않습니다.
!	대괄호로 묶이지 않은 임의의 문자와 같습니다.	b[!ae]ll은 bill과 bull을 찾으며 ball과 bell은 찾지 않습니다.
-	문자 범위 내에 있는 임의의 문자 한 개와 같습니다. Z에서 A가 아니라 A에서 Z까지 오름차순으로 범위를 지정해야 합니다.	b[a-c]d는 bad, bbd 및 bcd를 찾습니다.
#	임의의 숫자 문자 한 개와 같습니다.	1#3은 103, 113 및 123을 찾습니다.

3. 행 선택 방법

연산자

ANSI-92 와일드카드 문자

문자	설명	예제
%	임의의 수의 문자와 같습니다. 문자열 내의 원하는 위치에 별표(*)를 사용할 수 있습니다.	Wh%는 what, white 및 why를 찾으며 awhile 또는 watch는 찾지 않습니다.
_	임의의 알파벳 문자 한 개와 같습니다.	B_ll은 ball, bell 및 bill을 찾습니다.
[ ]	대괄호로 묶인 임의의 문자 한 개와 같습니다.	B[ae]ll은 ball과 bell을 찾으며 bill은 찾지 않습니다.
^	대괄호로 묶이지 않은 임의의 문자와 같습니다.	b[^ae]ll은 bill과 bull을 찾으며 ball과 bell은 찾지 않습니다.
-	문자 범위 내에 있는 임의의 문자 한 개와 같습니다. Z에서 A가 아니라 A에서 Z까지 오름차순으로 범위를 지정해야 합니다.	b[a-c]d는 bad, bbd 및 bcd를 찾습니다.

3. 행 선택 방법

연산자

LIKE 연산자 예

구 분	설 명
LIKE 'A%'	'A'로 시작하는 데이터만 검색
LIKE '%A'	'A'로 끝나는 테이터들만 검색
LIKE '%KIM%'	'KIM' 문자가 있는 데이터 들만 검색
LIKE '%K%I%'	'K' 문자와 'I'문자가 있는 데이터 들만 검색
LIKE '_A%'	'A'문자가 두 번째 위치한 데이터 들만 검색

3. 행 선택 방법

함수

문자 함수

함수	기능	구문	예제
Asc	문자열의 첫 번째 문자에 해당하는 문자 코드를 나타내는 정수를 반환한다.	Asc(*string*)	Asc("A") → 65 Asc("a") → 97. Asc("Apple") → 65.
Chr	지정된 문자 코드에 해당하는 문자를 반환한다.	Chr(*charcode*)	Chr(65) → A. Chr(97) → a.
Str	숫자를 문자로 변환하여 반환한다.	Str(*number*)	Str(459) → "459" Str(-45.6) → "-45.6" Str(45.01) → " 45.01".
Val	문자열에 포함된 숫자를 적절한 형식의 숫자 값으로 반환한다.	Val(*string*)	Val("2457") → 2457 Val(" 2 45 7") → 2457 Val("24 and 57") → 24.

3. 행 선택 방법

함수

문자 함수

함수	기능	구문	예제
Left	문자열의 왼쪽부터 지정된 수의 문자를 반환한다.	Left(*string, length*)	Str="Hello World" Left(Str, 1) → "H" Left(Str, 7) → "Hello W" Left(Str, 20) →"Hello World"
Right	문자열의 오른쪽부터 시작하여 지정된 수의 문자를 반환한다.	Right(*string, length*)	Right(Str, 1) → "d" Right(Str, 6) → " World" Right(Str, 20) →"Hello World"
Mid	문자열에서 첫 번째 지정된 수의 위치에서 두 번째 지정된 수의 문자를 반환한다.	Mid(*string, start* [, *length*])	Mid(Str, 2, 3) → "ello" Mid(Str, 2, 14) → "ello World" Mid(Str, 7) → "World"
Len	문자열의 문자 수를 반환한다.	Len(*string*)	Len("Ansan") → 5

3. 행 선택 방법

함수

문자 함수

함수	기능	구문	예제
Ltrim	지정한 문자열의 선행 공백을 제거한다.	LTrim(*string*)	MyString=" <-Trim-> " LTrim(MyString) → "<-Trim-> ".
RTrim	지정한 문자열의 후행 공백 제거한다.	RTrim(*string*)	RTrim(MyString) → " <-Trim->".
Trim	지정한 문자열의 선행/후행 공백 모두 제거한다.	Trim(*string*)	Trim(MyString) → "<-Trim->".
LCase	소문자로 변환된 문자열을 반환한다.	LCase(*string*)	Lcase("Hello World") → "hello world"
UCase	대문자로 변환된 문자열을 반환한다.	UCase(*string*)	UCase("Hello World") →"HELLO WORLD"

3. 행 선택 방법

함수

문자 함수 예제

Str="안산대학교인터넷정보과"

함수	결과
Left(Str, 2)	안산
Left(Str, 20)	안산대학교인터넷정보과
Right(Str, 3)	정보과
Right(Str, 15)	안산대학교인터넷정보과
Mid(Str, 3, 2)	대학
Mid(Str, 6, 3)	인터넷
Mid(Str, 6)	인터넷정보과
Len(Str)	11

3. 행 선택 방법

함수

날짜/시간 함수

함수	기능	구문	예제
Year	연도를 나타내는 정수를 반환한다.	Year(*date*)	Year(#2010-3-15#) →
Month	월을 나타내는 1에서 12 사이의 정수를 반환한다.	Month(*date*)	Month(#2010-3-15#) →
Day	날짜를 표시하는 1에서 31 사이의 정수를 반환한다.	Day(*date*)	Day(#2010-3-15#) →
Date	현재 시스템 날짜를 반환한다.	Date()	=Date() →
Now	컴퓨터의 시스템 날짜와 시간에 따라 현재 날짜와 시간을 반환한다.	Now()	=Now() →

3. 행 선택 방법

함수

날짜/시간 함수

함수	기능	구문	예제
Time	현재 시스템 시간을 반환한다..	Time	
Hour	하루 중 시를 나타내는 0에서 23 사이의 정수를 반환한다.	Hour(*time*)	Hour(#4:35:17 PM#) →
Minute	분을 나타내는 0에서 59 사이의 정수를 반환한다.	Minute(*time*)	Minute(#4:35:17 PM#) →

3. 행 선택 방법

함수

날짜/시간 함수

함수	기능	구문
DatePart	지정된 날짜의 특정 부분이 들어 있는 정수를 반환한다.	DatePart(*interval, date*)
DateAdd	지정된 시간 간격이 추가된 날짜를 반환한다.	DateAdd(*interval, number, date*)
DateDiff	지정된 두 날짜 간의 시간 간격 수를 반환한다.	DateDiff(*interval, date1, date2*)

설정	설명
yyyy	연도
q	분기
m	월
y	일년 중 일수
d	일

설정	설명
ww	주
h	시간
n	분
s	초

3. 행 선택 방법

함수

날짜/시간 함수 예제

(1) DatePart("q", #2010-4-5#) → 2
(2) DatePart("m", #2010-4-5#) → 4
(3) DatePart("d", #2010-3-15#) → 15
(4) DatePart("yyyy", #2010-3-5#) → 2010

(1) DateAdd("q", 1,#2010-3-5#) → 2
(2) DateAdd("m", 2,#2010-4-5#) → 6
(3) DateAdd("d", 5,#2010-3-15#) → 20
(4) DateAdd("yyyy", 2,#2010-3-5#) → 2012

3. 행 선택 방법

함수

날짜/시간 함수 예제

(1) DateDiff("q", #2010-9-5#, #2010-1-3#) → 2

(2) DateDiff ("m", #2010-9-5#, #2010-4-15#) → 5

(3) DateDiff ("d", #2010-3-15#, #2010-3-3#) → 12

(4) DateDiff ("yyyy", #2010-3-5#, #2008-9-5#) → 2

3. 행 선택 방법

함수

Choose Function

기능	값 목록 중에서 지정된 위치의 값을 반환한다.
구문	Choose (*position, value1, value2, ..., value_n*)
참고	• *position* 이 1보다 적으면 null 값을 반환한다. • *position* 이 값의 개수 보다 크면 null 값을 반환한다. • *position* 이 소수점이 있으면 정수만 취한다.

예제	반환 값	설명
Choose(1, "Tech", "on", "the", "Net")	"Tech"	값 목록 중 첫 번째 값을 반환한다.
Choose(2, "Tech", "on", "the", "Net")	"on"	값 목록 중 두 번째 값을 반환한다.
Choose(3, "Tech", "on", "the", "Net")	"the"	값 목록 중 세 번째 값을 반환한다.
Choose(4, "Tech", "on", "the", "Net")	"Net"	값 목록 중 네 번째 값을 반환한다.
Choose(5, "Tech", "on", "the", "Net")	NULL	값 목록의 개수를 초과하여 Null을 반환한다.
Choose(3.75, "Tech", "on", "the", "Net")	"the"	값 목록 중 세 번째(정수 부분) 값을 반환한다.

3. 행 선택 방법

함수

iif Function

기능	조건의 결과에 따라 참이나 거짓 중 하나의 값을 반환한다.
구문	iif (*condition, value_if_true, value_if_false*)

예제	반환 값	설명
iif ([Qty] >10, "large", "small")		[Qty]의 값이 10보다 크면 "large", 10이하이면 "small"을 반환한다.
iif ([지역]="서울" AND [부서]="영업부", "과장", "부장")		[지역]의 값이 "서울"이고 [부서]의 값이 "영업부"이면 "과장"을 그렇지 않으면 "부장"을 반환한다.

3. 행 선택 방법

함수

Switch Function

기능	목록을 평가하여 목록에서 True로 평가되는 첫 번째 식과 연결된 식 또는 Variant 값을 반환한다.
구문	Switch (*expression1, value1, expression2, value2, ... expression_n, value_n*)

예제	설명
Switch ([성별]=1, "남자", [성별]=2, "여자")	[성별]의 값이 1이면 "남자", 2이면 "여자"를 반환한다.
Switch ([수도]="서울","한국", [수도]="동경","일본",[수도]="북경","중국")	[수도]의 값 "서울"이면 "한국", "동경"이면 "일본", "북경"이면 "중국"을 반환한다.

3. 행 선택 방법

1. 비교 연산자를 사용하여 행 찾기

```
SELECT    ProductID, Name
FROM      Product
WHERE     Name = 'Blade' ;
```

```
SELECT   ProductID, Name
FROM     Product
WHERE    ProductID <= 12 ;
```

3. 행 선택 방법

1. 비교 + 논리 연산자를 사용하여 행 찾기

```
SELECT   ProductID, Name
FROM    Product
WHERE   ProductID = 2 OR ProductID = 4
            OR Name = 'Spokes' ;
```

```
SELECT   ProductID, Name
FROM    Product
WHERE   ProductID = 2 AND Color = “White”;
```

3. 행 선택 방법

값을 문자열의 일부로 포함하는 행 찾기

```
SELECT   ProductID, Name, Color
FROM    Production.Product
WHERE   Name LIKE ('%Frame%');
```

```
SELECT ProductID, Name, Color
FROM Production.Product
WHERE Name LIKE ('%Frame%')
AND Name LIKE ('HL%')
AND Color = 'Red' ;
```

3. 행 선택 방법

두 값 사이의 값을 가진 행 찾기

```
SELECT ProductID, Name, Color
FROM Production.Product
WHERE ProductID BETWEEN 725 AND 734;
```

값 목록에 있는 행 찾기

```
SELECT ProductID, Name, Color
FROM Production.Product
WHERE Name IN ('Blade', 'Crown Race', 'Spokes');
```

4. 데이터 정렬

```
SELECT        속성
FROM          테이블
[ WHERE       검색조건 ]
[ORDER BY    속성 [ASC|DESC] ];
```

- ✓ ORDER BY절에서 하나 이상의 속성을 사용하여 검색 결과를 정렬할 수 있음. 오름차순(디폴트): ASC / 내림차순: DESC
- ✓ ORDER BY절은 SELECT문에서 가장 마지막에 사용되는 절
- ✓ 널 값은 오름차순에서는 맨 마지막에 출력되고 내림차순에서는 맨 먼저 출력됨
- ✓ 여러 기준에 따라 정렬하려면 정렬 기준이 되는 속성을 차례대로 제시

4. 데이터 정렬

예제: 제품 테이블에서 제품이름과 단가를 검색하여 단가가 낮은 순서로 정렬한다

```
SELECT      제품이름, 단가
FROM        제품
ORDER BY    단가 ;
```

예제: 성적 테이블에서 학번, 성명, 국어, 영어, 수학, 총점을 검색하여 총점이 높은 순서로 정렬한다. 총점이 동점일 경우 국어 성적이 높은 순서로 정렬한다.

```
SELECT      학번, 성명, 국어, 영어, 수학, 총점
FROM        성적
ORDER BY    총점 DESC, 국어 DESC;
```

5. 데이터 그룹

집계 함수를 이용한 검색

- ✓ 특정 속성 값을 통계적으로 계산한 결과를 검색하기 위해 집계 함수(aggregate function)를 이용
- ✓ 집계 함수를 열 함수(column function)라고도 함
- ✓ 개수, 합계, 평균, 최대값, 최소값의 계산 기능을 제공
- ✓ 집계 함수 사용 시 주의 사항
  - 집계 함수는 널인 속성 값은 제외하고 계산함
  - 집계 함수는 WHERE 절에서는 사용할 수 없고 SELECT 절이나 HAVING 절에서만 사용 가능

5. 데이터 그룹

집계함수의 이해

- ✓ 집계 함수란 여러행 또는 테이블 전체 행으로부터 하나의 결과값을 반환하는 함수이다
- ✓ GROUP BY절을 이용하여 그룹 당 하나의 결과로 그룹화 할 수 있다.
- ✓ HAVING절을 사용하여 집계함수를 이용한 조건 비교를 할 수 있다.
- ✓ MIN, MAX 함수는 모든 자료형에 사용 할 수 있다.
- ✓ 일반적으로 가장 많이 사용하는 집계함수에는AVG(평균), COUNT(개수), MAX(최대값), MIN(최소값), SUM(합계) 등이 있다.

5. 데이터 그룹

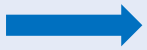
집계함수

함수	의미	사용 가능한 속성의 타입
COUNT	속성 값의 개수	모든 데이터
MAX	속성 값의 최대값	
MIN	속성 값의 최소값	
SUM	속성 값의 합계	숫자 데이터
AVG	속성 값의 평균	

5. 데이터 그룹

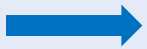
예제: 제품 테이블에서 모든 제품의 단가 평균을 검색

```
SELECT  AVG(단가)
FROM    제품;
```



(열 이름 없음)
₩27,803

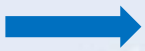
```
SELECT  AVG(단가) AS 단가의평균
FROM    제품;
```



단가의평균
₩27,803

예제: 제품 테이블에서 제품이름이 “대성 어묵”인 제품의 재고량 합계를 검색

```
SELECT  SUM(재고량) AS 재고량의합
FROM    제품
WHERE   제품이름="대성 어묵";
```

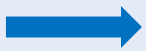


재고량의합
64

5. 데이터 그룹

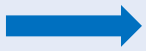
예제: 고객 테이블에서 등록된 모든 고객의 수를 검색

```
SELECT   COUNT(나이) AS "고객의 수"
FROM     고객;
```



고객의 수
123

```
SELECT   COUNT(*) AS "고객의 수"
FROM     고객;
```



고객의 수
130

❖ COUNT는 널인 속성 값은 제외하고 개수를 계산하므로 정확한 개수를 구할 경우 *를 사용한다.

예제: 제품 테이블에서 제조업체의 수를 검색

```
SELECT   COUNT(DISTINCT(제조업체수) AS "제조업체 수"
FROM     제품;
```

제조업체 수
5

5. 데이터 그룹

```
SELECT        속성
FROM          테이블
[ WHERE       검색조건 ]
[ GROUP BY   속성 ]
[ HAVING      검색조건 ]
[ORDER BY    속성 [ASC|DESC] ];
```

- ✓ GROUP BY 절을 이용해 특정 속성의 값이 같은 투플을 모아 그룹을 만들고, 그룹별로 검색
- ✓ GROUP BY 절에 그룹을 나누는 기준이 되는 속성을 지정
- ✓ HAVING 절을 이용해 그룹에 대한 조건을 작성
- ✓ 그룹을 나누는 기준이 되는 속성을 SELECT 절에도 작성하는 것이 좋음

5. 데이터 그룹

예제: 제품 테이블에서 제품분류와 제품분류의 평균 단가를 검색

```
SELECT  제품분류, AVG(단가)
FROM    제품;
```

→ 에러가 발생함

★ 집계 함수나 GROUP BY 절에 명시된 속성이 아닌 속성은 SELECT 절에 작성 불가

```
SELECT  제품분류, AVG(단가)
FROM    제품
GROUP BY 제품분류;
```

5. 데이터 그룹

예제: 주문 테이블에서 제품명별 수량의 합계를 검색

```
SELECT    제품명, SUM(수량) AS 총주문량
FROM      주문
GROUP BY 제품명;
```

제품명	총주문량
TV	24
냉장고	45
세탁기	32

예제: 제품 테이블에서 제조업체별로 제조한 제품의 개수와 제품 중 가장 비싼 단가와 가장 싼 단가를 검색

```
SELECT    제조업체, COUNT(*) AS 제품수, MAX(단가) AS 최고가
FROM      제품
GROUP BY 제조업체;
```

제조업체	제품수	최고가
태양식품	7	35000
현진무역	4	48000
진미식품	5	120000

5. 데이터 그룹

예제: 제품 테이블에서 제품을 3개 이상 제조한 제조업체별로 제품의 개수와, 제품 중 가장 비싼 단가를 검색

```
SELECT    제조업체, COUNT(*) AS 제품수, MAX(단가) AS 최고가
FROM      제품
GROUP BY  제조업체
HAVING    COUNT(*) >= 3;
```

예제: 제품 테이블에서 공급처가 "서울"인 제품을 3개 이상 제조한 제조업체별로 제품의 개수와, 제품 중 가장 최저 단가를 검색

```
SELECT    제조업체, COUNT(*) AS 제품수, MAX(단가) AS 최고가
FROM      제품
WHERE     공급처 = "서울"
GROUP BY  제조업체
HAVING    COUNT(*) >= 3;
```

수고했습니다.