SQL 중·상급 활용

2021. 10. 13.

Oracle SQL

Contents

도입부

머신러닝

2021.09.29

글로벌 칼럼 | '머신러닝은 만능이 아니다' ML 대신 SQL 쿼리를 써야하는 이유

Matt Asay | InfoWorld

때때로 머신러닝을 하는 가장 좋은 방법은 어떠한 머신러닝도 하지 않는 것이다. 아마존 응용 과학자 유진 얀은 "머신러닝의 첫 번째 규칙은 머신러닝 없이 시작하는 것"이라고 말했다. 무슨 뜻일까?

몇 달 동안 고된 노력을 투입해 머신러닝 모델을 공들여 만드는 것은 물론 멋진 일이다. 하지만 더 간단하고 편한 방법이 있을 때는 꼭 그렇게 할 필요가 없다.

지나친 단순화일 수도 있지만, 데이터 과학자 노아 로랑은 몇 년 전 "데이터 과학자는 대부분 산수만 한다"라고 말했다. 아주 틀린 말은 아니다. 데이터 연산 과정을 아무리 복잡하게 만들고 싶어도 시작은 간단한 것이 더 낫다는 점에서 얀과 로랑의 말은 정확하다.

1. 뉴스 기사

본문1

과장된 복잡함

데이터 과학자는 많은 급여를 받는다. 복잡한 전문 용어로 된 예측 분석과 다루기 힘든 모델로 고액 연봉을 정당화하고 싶어할지도 모른다. 하지만 그러지 말자. 데이터 과학에 대한 로랑의 몇 년 전 견해는 오늘날에도 유효하다. 로랑은 "기업이 겪고 있는 문제 가운데 머신러닝이 최고의 해결책이 되는 경우는 극히 일부다. 대부분은 좋은 데이터와 그 데이터가 의미하는 바만 이해하면 해결된다"라고 말했다. 로랑은 "SQL 쿼리로 데이터를 얻고, 백분위나 차이점 계산과 같은 기본적인 연산, 결과 그래프화, 결과 해석 혹은 개선사항 작성 등 간단한 방법을 추천한다"라고 덧붙였다.

아이로봇(iRobot) 데이터 과학자 브랜든 로러의 뻔뻔스러운 제안도 놀랄 일은 아니다. 로러는 "문제가 발생했을 때 두 가지 해결책을 만들자. 멀티클라우드 쿠버네티스에서 구동되는 심층 베이시안 트랜스포머(Bayesian transformer)와 매우 간소화된 조건으로 설정된 SQL 쿼리다. 전자는 이력서에 기재하고, 후자는 업무에 활용하자. 나도 회사도 행복해지는 비결이다"라고 설명했다.

본문2

데이터 이해가 출발점

먼저 얀은 결정적인 요소가 주어져도 데이터에서 의미를 도출하는 것이 얼마나 어려운 것인지 알아야 한다고 말했다. 얀은 "우선 데이터가 필요하다. 그 다음은 데이터 흐름을 이끌어줄 탄탄한 시스템이다. 대부분은 고품질 데이터 라벨링이 필요할 것이다"라고 설명했다.

다시 말해 시작부터 머신러닝 모델을 적용하는 것은 도움이 되지 않을 수 있다. 문제 해결을 시작할 때에는 데이터를 먼저 이해해야 한다. 수동으로, 혹은 실용적이거나 간단한 휴리스틱 방식으로 문제를 풀어나가는 것이다. 얀은 이를 깃허브 머신러닝 엔지니어 하멜 후세인의 말을 인용해 설명했다. 후세인은 "문제 상황과 데이터에 익숙해지는 것이 문제 해결의 출발점이다"라고 말했다.

예를 들어 표로 정리된 데이터를 처리할 때에는 통계를 돌리기 위한 데이터 표본이 시작점이다. 간단한 상관관계를 시작으로 산포도 작성 등 데이터를 시각화하는 것이다. 얀은 "제품을 추천하는 복잡한 머신러닝 모델을 설계하는 것보다 전년도 인기 제품을 추천하는 것이 낫다"라고 말했다. 결과에서 패턴을 찾는 것은 나중 일이다. 이 방법은 머신러닝 개발자가 데이터와 더 친해지는 방법이며, 필요에 따라 더 좋은 머신러닝 모델을 개발하는 데 도움이 된다.

마무리

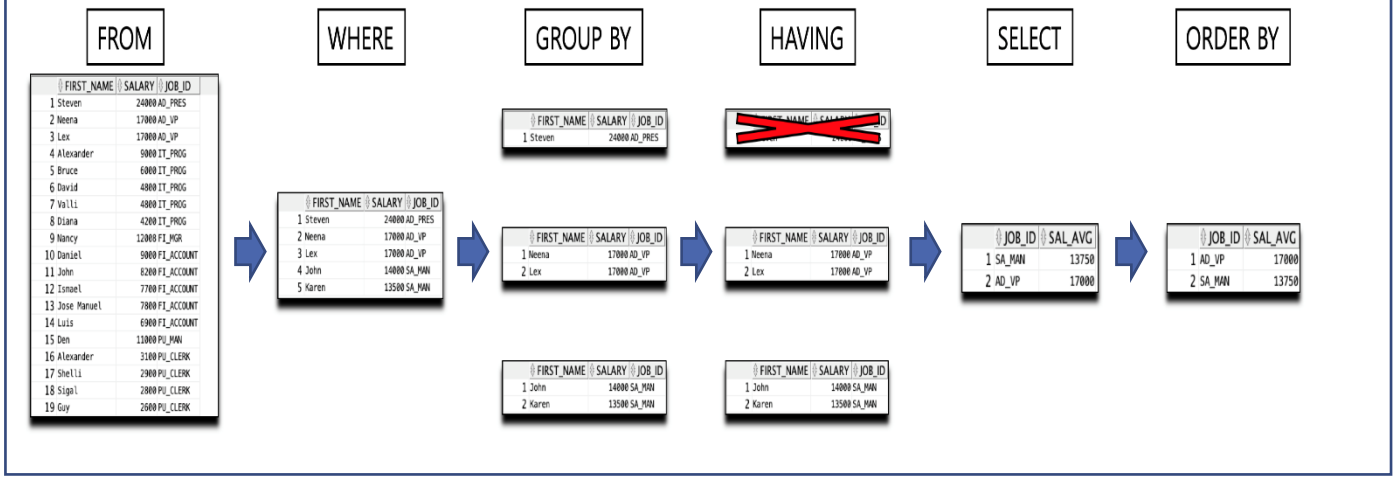
복잡한 머신러닝을 개발하고 싶은 마음이 생길 수 있다. 하지만 데이터 과학자가 지녀야 할 가장 중요한 덕목은 머신러닝 대신 회귀 분석이나 조건문을 이용해야 할 때를 아는 것이다.

1. SELECT절 작성과 SELECT절 실행 순서

SELECT절 작성

```sql
SELECT
    JOB_ID
    ,AVG(SALARY) SAL_AVG
FROM
    EMPLOYEES
WHERE
    SALARY > 13000
GROUP BY
    JOB_ID
HAVING
    COUNT(*) > 1
ORDER BY SAL_AVG DESC;
```

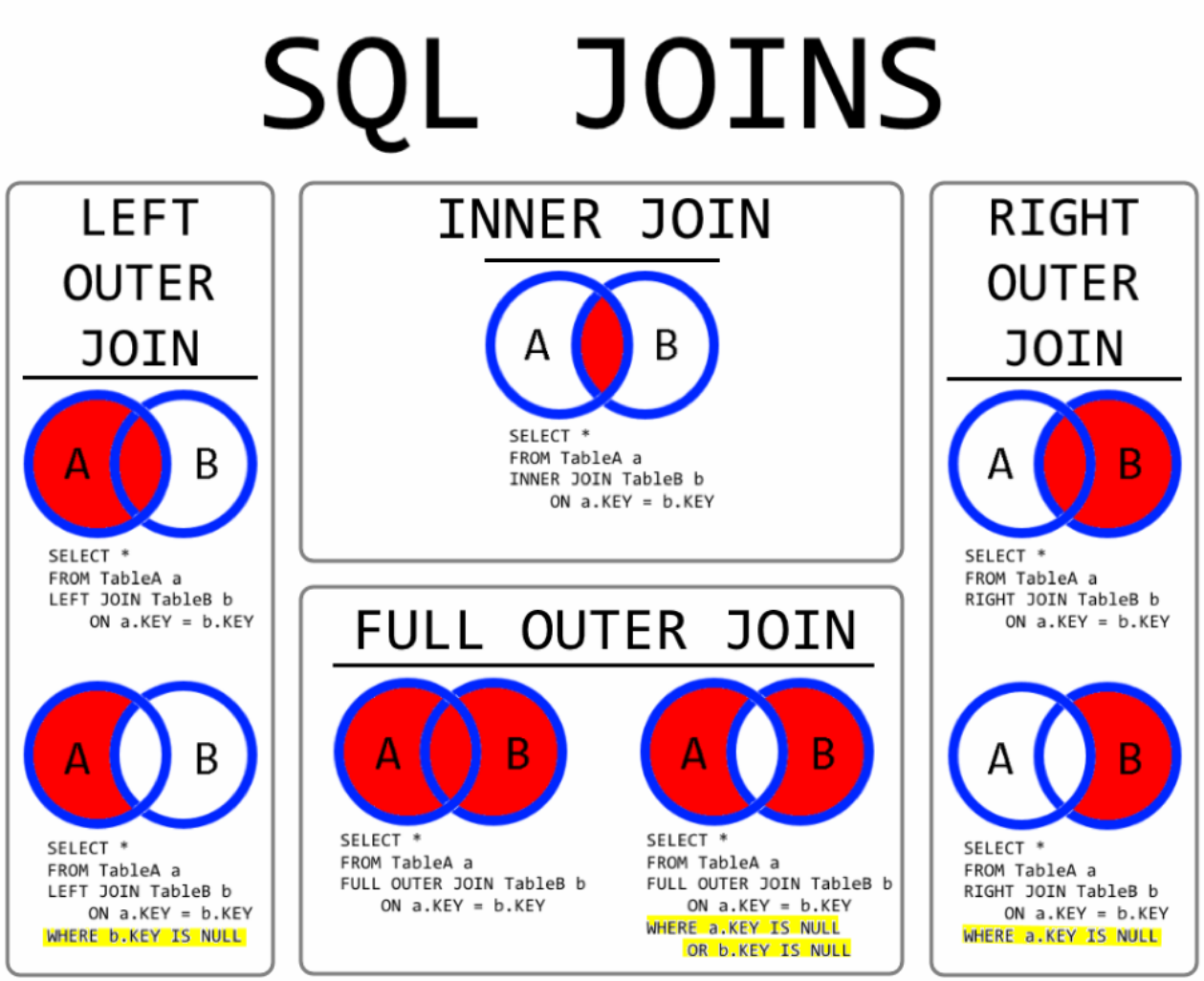
SELECT절 실행 순서



SELECT절 실행 순서대로

SELECT절을 해석하거나 작성해야 함

2. JOIN 종류



- INNER JOIN과 LEFT OUTER JOIN을 가장 많이 활용함

3. INNER JOIN

EMP

```
SELECT *
FROM SYSTEM.EMP;
```

	ENO	ENAME	SEX	JOB	MGR	HDATE	SAL	COMM	DNO
1	0001	안영희	여	경영	(null)	91/01/01	4800	0	01
2	0201	안영숙	여	지원	0001	91/02/01	3900	2000	01
3	0202	손하늘	여	지원	0001	91/12/01	3510	980	01
4	0301	이승철	남	회계	0001	92/02/01	3400	0	02
5	0302	박선경	여	회계	0301	91/03/02	3300	0	02
6	1001	문시현	남	모델링	0201	91/02/01	4500	520	10
7	1002	김주란	여	모델링	0201	92/03/03	4100	330	20
8	1003	양선호	남	모델링	0201	95/02/21	4300	(null)	30
9	2001	남궁연호	남	개발	0202	93/12/13	3950	200	10
10	2002	제갈민	남	개발	0202	96/04/30	1520	2000	20
11	2003	정의찬	남	개발	0202	92/03/03	4350	(null)	30
12	2007	이초록	남	개발	0001	92/09/05	1989	2300	30
13	2008	윤고은	여	개발	0001	92/03/03	2100	(null)	40
14	3002	권아현	여	분석	1002	01/01/29	2900	(null)	20
15	0309	김선유	남	회계	0302	11/01/03	900	90	02
16	3001	김선유	남	분석	1001	98/10/17	3200	300	10
17	0702	김민지	남	회계	0301	17/01/09	1100	60	02
18	0269	권나현	여	분석	0301	15/05/21	2600	1900	10
19	0401	김진성	남	회계	1001	08/03/13	3200	1000	10
20	0801	천유정	여	분석	1001	00/10/09	2900	600	02
21	0120	김경현	남	지원	1002	99/09/05	4000	2500	20
22	0111	최우석	남	분석	1002	99/09/17	5000	2500	90

DEPT

```
SELECT *
FROM SYSTEM.DEPT;
```

	DNO	DNAME	LOC
1	01	총무	서울
2	02	회계	서울
3	10	ERP	서울
4	20	ISP	부산
5	30	ITEA	광주
6	40	CRM	대전
7	50	POS	(null)

두 테이블을 INNER JOIN한다면?

3. INNER JOIN

JOIN 실시

```
SELECT B.ENO,
       B.ENAME,
       B.SEX,
       B.JOB,
       B.SAL,
       A.DNO,
       A.DNAME
FROM SYSTEM.DEPT A,
     SYSTEM.EMP B
WHERE A.DNO = B.DNO;
```

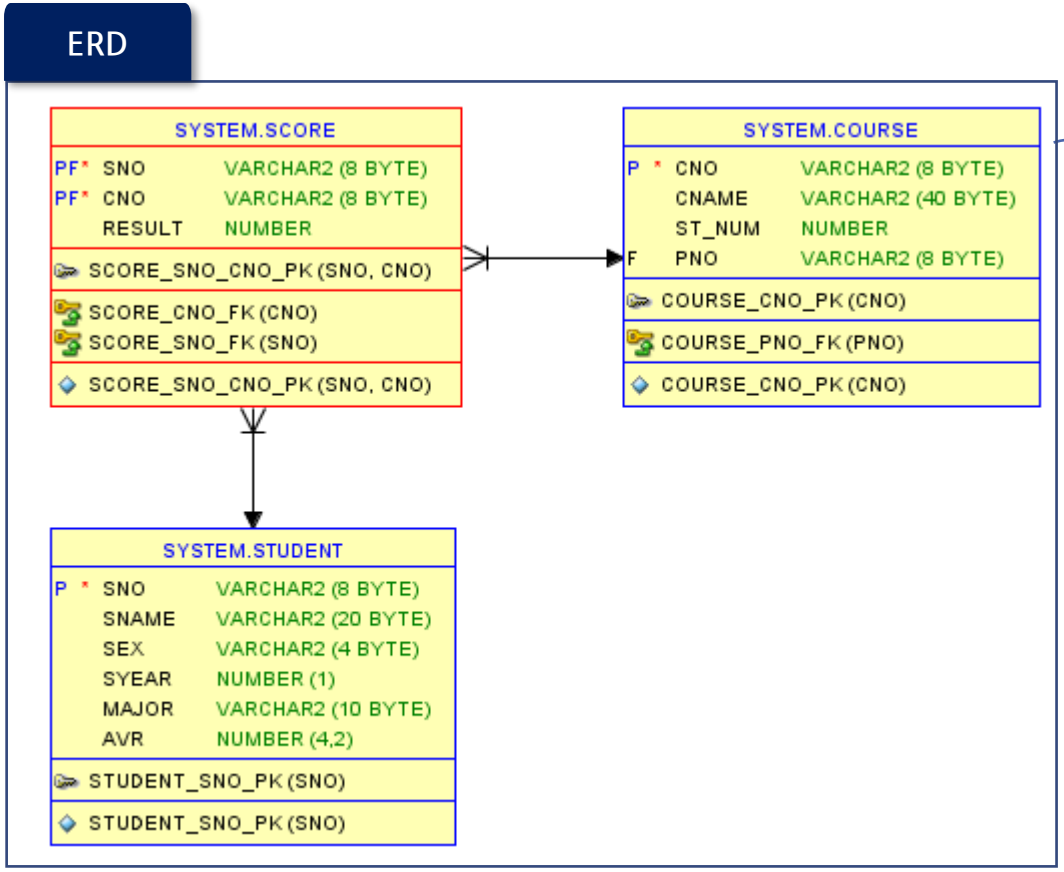
SYSTEM : 'SYSTEM'이라는 계정명
A : 'DEPT'테이블에 대한 별칭

	ENO	ENAME	SEX	JOB	SAL	DNO	DNAME
1	0001	안영희	여	경영	4800	01	총무
2	0201	안영숙	여	지원	3900	01	총무
3	0202	손하늘	여	지원	3510	01	총무
4	0301	이승철	남	회계	3400	02	회계
5	0302	박선경	여	회계	3300	02	회계
6	1001	문시현	남	모델링	4500	10	ERP
7	1002	김주란	여	모델링	4100	20	ISP
8	1003	양선호	남	모델링	4300	30	ITEA
9	2001	남궁연호	남	개발	3950	10	ERP
10	2002	제갈민	남	개발	1520	20	ISP
11	2003	정의찬	남	개발	4350	30	ITEA
12	2007	이초록	남	개발	1989	30	ITEA
13	2008	윤고은	여	개발	2100	40	CRM
14	3002	권아현	여	분석	2900	20	ISP
15	0309	김선유	남	회계	900	02	회계
16	3001	김선유	남	분석	3200	10	ERP
17	0702	김민지	남	회계	1100	02	회계
18	0269	권나현	여	분석	2600	10	ERP
19	0401	김진성	남	회계	3200	10	ERP
20	0801	천유정	여	분석	2900	02	회계
21	0120	김경현	남	지원	4000	20	ISP

- 조인 조건 : A.DNO = B.DNO
  (INNER JOIN에 대한 조인 조건은 WHERE절에 기입)
- 두 테이블에 모두 존재하는 레코드만 결과로 산출함
- 이에 따라, ENAME이 '최우석'인 레코드는 결과로 산출되지 않음
- **만약, ENAME이 '최우석'인 레코드가 결과에 산출되도록 하려면, 어떻게 해야 할까?**

3. INNER JOIN

ERD



- 세 테이블을 INNER JOIN하여, '학생별&과목별 점수'결과를 조회하고 싶다면?
  (결과에는 SNAME, CNAME, RESULT 세 컬럼만 존재해야 함)

3. INNER JOIN

JOIN 결과

```
SELECT A.SNAME,
       C.CNAME,
       B.RESULT
FROM SYSTEM.STUDENT A,
     SYSTEM.SCORE B,
     SYSTEM.COURSE C
WHERE A.SNO = B.SNO
      AND B.CNO = C.CNO
ORDER BY A.SNAME;
```

	SNAME	CNAME	RESULT
1	갑서진	일반화학실험	47
2	갑서진	유전체학	97
3	갑서진	식품분석실험	61
4	갑서진	열역학	65
5	갑서진	고분자화학	76
6	갑서진	식품화학	65
7	갑서진	환경화학	42
8	갑서진	위상수학	53
9	갑서진	분류학실험	40
10	갑서진	무기화학	68
11	갑서진	생리학	44
12	갑서진	실험물리학	78
13	갑서진	영양생리학	95
14	갑서진	식물학	55
15	갑서진	유기물리학	70

- 조인 조건 : A.SNO = B.SNO
  B.CNO = C.CNO
- 세 테이블에 모두 존재하는 레코드만 결과로 산출함
- **FK를 파악하여, 세 테이블을 연결하면 됨**

4. LEFT OUTER JOIN

EMP

```
SELECT *
FROM SYSTEM.EMP;
```

	ENO	ENAME	SEX	JOB	MGR	HDATE	SAL	COMM	DNO
1	0001	안영희	여	경영	(null)	91/01/01	4800	0	01
2	0201	안영숙	여	지원	0001	91/02/01	3900	2000	01
3	0202	손하늘	여	지원	0001	91/12/01	3510	980	01
4	0301	이승철	남	회계	0001	92/02/01	3400	0	02
5	0302	박선경	여	회계	0301	91/03/02	3300	0	02
6	1001	문시현	남	모델링	0201	91/02/01	4500	520	10
7	1002	김주란	여	모델링	0201	92/03/03	4100	330	20
8	1003	양선호	남	모델링	0201	95/02/21	4300	(null)	30
9	2001	남궁연호	남	개발	0202	93/12/13	3950	200	10
10	2002	제갈민	남	개발	0202	96/04/30	1520	2000	20
11	2003	정의찬	남	개발	0202	92/03/03	4350	(null)	30
12	2007	이초록	남	개발	0001	92/09/05	1989	2300	30
13	2008	윤고은	여	개발	0001	92/03/03	2100	(null)	40
14	3002	권아현	여	분석	1002	01/01/29	2900	(null)	20
15	0309	김선유	남	회계	0302	11/01/03	900	90	02
16	3001	김선유	남	분석	1001	98/10/17	3200	300	10
17	0702	김민지	남	회계	0301	17/01/09	1100	60	02
18	0269	권나현	여	분석	0301	15/05/21	2600	1900	10
19	0401	김진성	남	회계	1001	08/03/13	3200	1000	10
20	0801	천유정	여	분석	1001	00/10/09	2900	600	02
21	0120	김경현	남	지원	1002	99/09/05	4000	2500	20
22	0111	최우석	남	분석	1002	99/09/17	5000	2500	90

DEPT

```
SELECT *
FROM SYSTEM.DEPT;
```

	DNO	DNAME	LOC
1	01	총무	서울
2	02	회계	서울
3	10	ERP	서울
4	20	ISP	부산
5	30	ITEA	광주
6	40	CRM	대전
7	50	POS	(null)

두 테이블을 LEFT OUTER JOIN한다면?

4. LEFT OUTER JOIN

JOIN 실시

```
SELECT A.ENO,
       A.ENAME,
       A.SEX,
       A.JOB,
       A.SAL,
       B.DNO,
       B.DNAME
FROM SYSTEM.EMP A
LEFT JOIN
SYSTEM.DEPT B
ON (A.DNO = B.DNO);
```

	ENO	ENAME	SEX	JOB	SAL	DNO	DNAME
1	0001	안영희	여	경영	4800	01	총무
2	0201	안영숙	여	지원	3900	01	총무
3	0202	손하늘	여	지원	3510	01	총무
4	0301	이승철	남	회계	3400	02	회계
5	0302	박선경	여	회계	3300	02	회계
6	0309	김선유	남	회계	900	02	회계
7	0702	김민지	남	회계	1100	02	회계
8	0801	천유정	여	분석	2900	02	회계
9	1001	문시현	남	모델링	4500	10	ERP
10	2001	남궁연호	남	개발	3950	10	ERP
11	3001	김선유	남	분석	3200	10	ERP
12	0269	권나현	여	분석	2600	10	ERP
13	0401	김진성	남	회계	3200	10	ERP
14	1002	김주란	여	모델링	4100	20	ISP
15	2002	제갈민	남	개발	1520	20	ISP
16	3002	권아현	여	분석	2900	20	ISP
17	0120	김경현	남	지원	4000	20	ISP
18	1003	양선호	남	모델링	4300	30	ITEA
19	2003	정의찬	남	개발	4350	30	ITEA
20	2007	이초록	남	개발	1989	30	ITEA
21	2008	윤고은	여	개발	2100	40	CRM
22	0111	최우석	남	분석	5000	(null)	(null)

- 조인 조건 : A.DNO = B.DNO
  (LEFT OUTER JOIN에 대한 조인 조건은 ON절에 기입)
- LEFT에 해당하는 'EMP'테이블 내 모든 레코드가 조회됨
- 이에 따라, ENAME이 '최우석'인 레코드까지 결과로 산출됨
- 다만, 해당 레코드의 DNO컬럼값과 DNAME 컬럼값은 존재하지 않기 때문에, 해당 컬럼값은 NULL로 나타남
- **NULL값을 생성시키지 않는 INNER JOIN만 사용해도 될 것 같은데, 굳이 LEFT OUTER JOIN을 배우는 이유는 무엇일까?**

5. GROUP BY

EMP

```
SELECT *
FROM SYSTEM.EMP;
```

	ENO	ENAME	SEX	JOB	MGR	HDATE	SAL	COMM	DNO
1	0001	안영희	여	경영	(null)	91/01/01	4800	0	01
2	0201	안영숙	여	지원	0001	91/02/01	3900	2000	01
3	0202	손하늘	여	지원	0001	91/12/01	3510	980	01
4	0301	이승철	남	회계	0001	92/02/01	3400	0	02
5	0302	박선경	여	회계	0301	91/03/02	3300	0	02
6	1001	문시현	남	모델링	0201	91/02/01	4500	520	10
7	1002	김주란	여	모델링	0201	92/03/03	4100	330	20
8	1003	양선호	남	모델링	0201	95/02/21	4300	(null)	30
9	2001	남궁연호	남	개발	0202	93/12/13	3950	200	10
10	2002	제갈민	남	개발	0202	96/04/30	1520	2000	20
11	2003	정의찬	남	개발	0202	92/03/03	4350	(null)	30
12	2007	이초록	남	개발	0001	92/09/05	1989	2300	30
13	2008	윤고은	여	개발	0001	92/03/03	2100	(null)	40
14	3002	권아현	여	분석	1002	01/01/29	2900	(null)	20
15	0309	김선유	남	회계	0302	11/01/03	900	90	02
16	3001	김선유	남	분석	1001	98/10/17	3200	300	10
17	0702	김민지	남	회계	0301	17/01/09	1100	60	02
18	0269	권나현	여	분석	0301	15/05/21	2600	1900	10
19	0401	김진성	남	회계	1001	08/03/13	3200	1000	10
20	0801	천유정	여	분석	1001	00/10/09	2900	600	02
21	0120	김경현	남	지원	1002	99/09/05	4000	2500	20
22	0111	최우석	남	분석	1002	99/09/17	5000	2500	90

DEPT

```
SELECT *
FROM SYSTEM.DEPT;
```

	DNO	DNAME	LOC
1	01	총무	서울
2	02	회계	서울
3	10	ERP	서울
4	20	ISP	부산
5	30	ITEA	광주
6	40	CRM	대전
7	50	POS	(null)

두 테이블을 INNER JOIN한 후,
부서별 SALARY합계를 조회하려면
어떻게 할까?

(단, 결과에 'DNO'컬럼과 'DNAME'컬럼 둘 다 나타나야 함)

5. GROUP BY

GROUP BY 실시

```
SELECT B.DNO,
       B.DNAME,
       SUM(A.SAL) AS SAL_SUM
FROM SYSTEM.EMP A,
     SYSTEM.DEPT B
WHERE A.DNO = B.DNO
GROUP BY B.DNO, B.DNAME;
```

	DNO	DNAME	SAL_SUM
1	20	ISP	12520
2	40	CRM	2100
3	30	ITEA	10639
4	10	ERP	17450
5	01	총무	12210
6	02	회계	11600

- GROUP BY 기준 : DNO, DNAME
- 조회 결과에 DNO컬럼과 DNAME컬럼이 동시에 나타나도록 명시했기 때문에, 해당 두 컬럼을 GROUP BY기준으로 선정해야 함
- GROUP BY 동작 방식 :
  1. GROUP BY절에 기입된 컬럼내 값 별로 레코드를 묶음
  2. 묶여진 레코드 별로 SELECT절에 기입된 집계함수 연산 실시
- GROUP BY를 통해 SUM()뿐만 아니라, AVG(), COUNT() 등의 다른 집계 연산도 수행할 수 있음
- **GROUP BY와 WINDOW함수의 PARTITION BY의 차이를 반드시 이해해야 함**

6. ORDER BY

GROUP BY 결과

```
SELECT B.DNO,
       B.DNAME,
       SUM(A.SAL) AS SAL_SUM
FROM SYSTEM.EMP A,
SYSTEM.DEPT B
WHERE A.DNO = B.DNO
GROUP BY B.DNO, B.DNAME;
```

	DNO	DNAME	SAL_SUM
1	20	ISP	12520
2	40	CRM	2100
3	30	ITEA	10639
4	10	ERP	17450
5	01	총무	12210
6	02	회계	11600

해당 GROUP BY결과를 DNO컬럼으로 정렬하여 보여주는 방법은?

6. ORDER BY

ORDER BY 오름차순

```
SELECT B.DNO,
       B.DNAME,
       SUM(A.SAL) AS SAL_SUM
FROM SYSTEM.EMP A,
SYSTEM.DEPT B
WHERE A.DNO = B.DNO
GROUP BY B.DNO, B.DNAME
ORDER BY B.DNO;
```

	DNO	DNAME	SAL_SUM
1	01	총무	12210
2	02	회계	11600
3	10	ERP	17450
4	20	ISP	12520
5	30	ITEA	10639
6	40	CRM	2100

ORDER BY 내림차순

```
SELECT B.DNO,
       B.DNAME,
       SUM(A.SAL) AS SAL_SUM
FROM SYSTEM.EMP A,
SYSTEM.DEPT B
WHERE A.DNO = B.DNO
GROUP BY B.DNO, B.DNAME
ORDER BY B.DNO DESC;
```

	DNO	DNAME	SAL_SUM
1	40	CRM	2100
2	30	ITEA	10639
3	20	ISP	12520
4	10	ERP	17450
5	02	회계	11600
6	01	총무	12210

- ORDER BY B.DNO : DNO컬럼 기준으로 오름차순 정렬 실시
- ORDER BY B.DNO DESC : DNO컬럼 기준으로 내림차순 정렬 실시

LEFT JOIN 결과

```
SELECT A.ENO,
       A.ENAME,
       A.SEX,
       A.JOB,
       A.SAL,
       B.DNO,
       B.DNAME
FROM SYSTEM.EMP A
LEFT JOIN
SYSTEM.DEPT B
ON (A.DNO = B.DNO);
```

	ENO	ENAME	SEX	JOB	SAL	DNO	DNAME
1	0001	안영희	여	경영	4800	01	총무
2	0201	안영숙	여	지원	3900	01	총무
3	0202	손하늘	여	지원	3510	01	총무
4	0301	이승철	남	회계	3400	02	회계
5	0302	박선경	여	회계	3300	02	회계
6	0309	김선유	남	회계	900	02	회계
7	0702	김민지	남	회계	1100	02	회계
8	0801	천유정	여	분석	2900	02	회계
9	1001	문시현	남	모델링	4500	10	ERP
10	2001	남궁연호	남	개발	3950	10	ERP
11	3001	김선유	남	분석	3200	10	ERP
12	0269	권나현	여	분석	2600	10	ERP
13	0401	김진성	남	회계	3200	10	ERP
14	1002	김주란	여	모델링	4100	20	ISP
15	2002	제갈민	남	개발	1520	20	ISP
16	3002	권아현	여	분석	2900	20	ISP
17	0120	김경현	남	지원	4000	20	ISP
18	1003	양선호	남	모델링	4300	30	ITEA
19	2003	정의찬	남	개발	4350	30	ITEA
20	2007	이초록	남	개발	1989	30	ITEA
21	2008	윤고은	여	개발	2100	40	CRM
22	0111	최우석	남	분석	5000	(null)	(null)

- 앞서 LEFT JOIN를 실시했을 시, ENAME이 '최우석'인 레코드의 DNO컬럼과 DNAME컬럼에는 NULL이 입력되었음
- NULL대신 'EMPTY'라는 값을 입력시키려면?

CASE 사용 결과

```
SELECT A.ENO,
       A.ENAME,
       A.SEX,
       A.JOB,
       A.SAL,
       CASE
            WHEN B.DNO IS NULL THEN 'EMPTY'
            ELSE B.DNO
       END AS DNO,
       CASE
            WHEN B.DNAME IS NULL THEN 'EMPTY'
            ELSE B.DNAME
       END AS DNAME
FROM SYSTEM.EMP A
LEFT JOIN
SYSTEM.DEPT B
ON (A.DNO = B.DNO);
```

	ENO	ENAME	SEX	JOB	SAL	DNO	DNAME
1	0001	안영희	여	경영	4800	01	총무
2	0201	안영숙	여	지원	3900	01	총무
3	0202	손하늘	여	지원	3510	01	총무
4	0301	이승철	남	회계	3400	02	회계
5	0302	박선경	여	회계	3300	02	회계
6	0309	김선유	남	회계	900	02	회계
7	0702	김민지	남	회계	1100	02	회계
8	0801	천유정	여	분석	2900	02	회계
9	1001	문시현	남	모델링	4500	10	ERP
0	2001	남궁연호	남	개발	3950	10	ERP
1	3001	김선유	남	분석	3200	10	ERP
2	0269	권나현	여	분석	2600	10	ERP
3	0401	김진성	남	회계	3200	10	ERP
4	1002	김주란	여	모델링	4100	20	ISP
5	2002	제갈민	남	개발	1520	20	ISP
6	3002	권아현	여	분석	2900	20	ISP
7	0120	김경현	남	지원	4000	20	ISP
8	1003	양선호	남	모델링	4300	30	ITEA
9	2003	정의찬	남	개발	4350	30	ITEA
20	2007	이초록	남	개발	1989	30	ITEA
21	2008	윤고은	여	개발	2100	40	CRM
22	0111	최우석	남	분석	5000	EMPTY	EMPTY

- CASE 구문은 python의 IF구문과 동일함
- IF = WHEN, ELIF = WHEN, ELSE = ELSE
- **CASE 구문에 의해 기존 테이블 컬럼에 변형이 발생하지 않고, 해당 CASE 구문이 적용된 새로운 벡터가 생성됨 (SQL 함수 반환값도 이와 동일함)**

STUDENT 테이블

```
SELECT *
FROM SYSTEM.STUDENT;
```

	SNO	SNAME	SEX	SYEAR	MAJOR	AVR
1	915301	정동상	남	4	화학	0.95
2	905301	유태지	남	4	화학	3.28
3	905302	정욱상	남	4	화학	1.44
4	915303	정욱주	남	4	화학	0.95
5	923903	정남윤	남	3	생물	3.23
6	923904	한현석	남	3	생물	2.45
7	933901	김용서	남	2	생물	1.48
8	915304	권보수	남	4	화학	2.32
9	915305	최정희	여	3	화학	0.58
10	948203	황보우리	여	1	식영	2.83
11	948204	서창동	남	1	식영	3.21
12	925306	김재백	남	2	화학	2.78
13	933902	김현수	남	2	생물	2.48
14	933903	정승동	남	2	생물	2.99
15	935303	김완창	남	2	화학	0.34
16	945302	김람석	남	1	화학	3.56
17	945303	남궁경아	여	1	화학	2.36
18	945314	이철윤	남	1	화학	2.22
19	913901	황수현	남	4	생물	2.34
20	913902	황진혜	여	4	생물	3.15

CASE문을 사용하여, 4학년인 학생과 4학년이 아닌 학생을 구분하는 컬럼을 어떻게 생성할까?

CASE 사용 결과

```
SELECT SNO,
       SNAME,
       SEX,
       SYEAR,
       CASE
            WHEN SYEAR = 4 THEN 1
            ELSE 0
       END AS IS_SENIOR,
       MAJOR,
       AVR
FROM SYSTEM.STUDENT;
```

	SNO	SNAME	SEX	SYEAR	IS_SENIOR	MAJOR	AVR
1	915301	정동상	남	4	1	화학	0.95
2	905301	유태지	남	4	1	화학	3.28
3	905302	정욱상	남	4	1	화학	1.44
4	915303	정욱주	남	4	1	화학	0.95
5	923903	정남윤	남	3	0	생물	3.23
6	923904	한현석	남	3	0	생물	2.45
7	933901	김용서	남	2	0	생물	1.48
8	915304	권보수	남	4	1	화학	2.32
9	915305	최정희	여	3	0	화학	0.58
10	948203	황보우리	여	1	0	식영	2.83
11	948204	서창동	남	1	0	식영	3.21
12	925306	김재백	남	2	0	화학	2.78

- 이처럼, CASE구문을 사용하여 BOOLEAN형 컬럼을 생성하는 경우도 많음
- **BOOLEAN형 컬럼을 포함한 쿼리를 INLINE VIEW로 작성하고,**
  메인 쿼리에서 BOOLEAN형 컬럼값이 1인 레코드만 추출하는 로직을 자주 사용함

STUDENT 테이블

```
SELECT *
FROM SYSTEM.STUDENT;
```

	SNO	SNAME	SEX	SYEAR	MAJOR	AVR
1	915301	정동상	남	4	화학	0.95
2	905301	유태지	남	4	화학	3.28
3	905302	정욱상	남	4	화학	1.44
4	915303	정욱주	남	4	화학	0.95
5	923903	정남윤	남	3	생물	3.23
6	923904	한현석	남	3	생물	2.45
7	933901	김용서	남	2	생물	1.48
8	915304	권보수	남	4	화학	2.32
9	915305	최정희	여	3	화학	0.58
10	948203	황보우리	여	1	식영	2.83
11	948204	서창동	남	1	식영	3.21
12	925306	김재백	남	2	화학	2.78
13	933902	김현수	남	2	생물	2.48
14	933903	정승동	남	2	생물	2.99
15	935303	김완창	남	2	화학	0.34
16	945302	김람석	남	1	화학	3.56
17	945303	남궁경아	여	1	화학	2.36
18	945314	이철윤	남	1	화학	2.22
19	913901	황수현	남	4	생물	2.34
20	913902	황진혜	여	4	생물	3.15

CASE문을 사용하여, SYEAR를 한글로 나타내는 컬럼을 어떻게 생성할까?

CASE 사용 결과

```
SELECT SNO,
       SNAME,
       SEX,
       CASE
            WHEN SYEAR = 1 THEN '1학년'
            WHEN SYEAR = 2 THEN '2학년'
            WHEN SYEAR = 3 THEN '3학년'
            WHEN SYEAR = 4 THEN '4학년'
            ELSE '고학년'
       END AS SYEAR_STR,
       MAJOR,
       AVR
FROM SYSTEM.STUDENT;
```

	SNO	SNAME	SEX	SYEAR_STR	MAJOR	AVR
1	915301	정동상	남	4학년	화학	0.95
2	905301	유태지	남	4학년	화학	3.28
3	905302	정욱상	남	4학년	화학	1.44
4	915303	정욱주	남	4학년	화학	0.95
5	923903	정남윤	남	3학년	생물	3.23
6	923904	한현석	남	3학년	생물	2.45
7	933901	김용서	남	2학년	생물	1.48
8	915304	권보수	남	4학년	화학	2.32
9	915305	최정희	여	3학년	화학	0.58
10	948203	황보우리	여	1학년	식영	2.83
11	948204	서창동	남	1학년	식영	3.21
12	925306	김재백	남	2학년	화학	2.78
13	933902	김현수	남	2학년	생물	2.48
14	933903	정승동	남	2학년	생물	2.99
15	935303	김완창	남	2학년	화학	0.34
16	945302	김람석	남	1학년	화학	3.56
17	945303	남궁경아	여	1학년	화학	2.36
18	945314	이철윤	남	1학년	화학	2.22

2. ||(CONCAT)

STUDENT 테이블

```
SELECT *
FROM SYSTEM.STUDENT;
```

	SNO	SNAME	SEX	SYEAR	MAJOR	AVR
1	915301	정동상	남	4	화학	0.95
2	905301	유태지	남	4	화학	3.28
3	905302	정욱상	남	4	화학	1.44
4	915303	정욱주	남	4	화학	0.95
5	923903	정남윤	남	3	생물	3.23
6	923904	한현석	남	3	생물	2.45
7	933901	김용서	남	2	생물	1.48
8	915304	권보수	남	4	화학	2.32
9	915305	최정희	여	3	화학	0.58
10	948203	황보우리	여	1	식영	2.83
11	948204	서창동	남	1	식영	3.21
12	925306	김재백	남	2	화학	2.78
13	933902	김현수	남	2	생물	2.48
14	933903	정승동	남	2	생물	2.99
15	935303	김완창	남	2	화학	0.34
16	945302	김람석	남	1	화학	3.56
17	945303	남궁경아	여	1	화학	2.36
18	945314	이철윤	남	1	화학	2.22
19	913901	황수현	남	4	생물	2.34
20	913902	황진혜	여	4	생물	3.15

CASE구문을 사용하지 않고, SYEAR컬럼 값 뒤에 '학년'문자열만 붙여도 되지 않는가?

2. ||(CONCAT)

||(CONCAT)

```
SELECT SNO,
       SNAME,
       SEX,
       TO_CHAR(SYEAR) || '학년' AS SYEAR_STR,
       MAJOR,
       AVR
FROM SYSTEM.STUDENT;
```

	SNO	SNAME	SEX	SYEAR_STR	MAJOR	AVR
1	915301	정동상	남	4학년	화학	0.95
2	905301	유태지	남	4학년	화학	3.28
3	905302	정욱상	남	4학년	화학	1.44
4	915303	정욱주	남	4학년	화학	0.95
5	923903	정남윤	남	3학년	생물	3.23
6	923904	한현석	남	3학년	생물	2.45
7	933901	김용서	남	2학년	생물	1.48
8	915304	권보수	남	4학년	화학	2.32
9	915305	최정희	여	3학년	화학	0.58
10	948203	황보우리	여	1학년	식영	2.83
11	948204	서창동	남	1학년	식영	3.21
12	925306	김재백	남	2학년	화학	2.78
13	933902	김현수	남	2학년	생물	2.48
14	933903	정승동	남	2학년	생물	2.99

- CONCAT()은 두 개 이상의 문자열을 연결시키는 함수
- '||'키워드는 CONCAT()함수와 동일한 효과를 가짐
- CONCAT()대신 '||'를 주로 사용함

3. SUBSTR()

가상테이블

```
SELECT *
FROM (
    SELECT '20210101' AS DATE_CHAR
    FROM DUAL
    UNION ALL
    SELECT '20220205' AS DATE_CHAR
    FROM DUAL
    UNION ALL
    SELECT '20230315' AS DATE_CHAR
    FROM DUAL
    UNION ALL
    SELECT '20240420' AS DATE_CHAR
    FROM DUAL
    UNION ALL
    SELECT '20250530' AS DATE_CHAR
    FROM DUAL
)
;
```

	DATE_CHAR
1	20210101
2	20220205
3	20230315
4	20240420
5	20250530

해당 테이블에서 YEAR, MONTH, DATE컬럼을 생성하는 방법은?

3. SUBSTR()

SUBSTR()

```
SELECT SUBSTR(DATE_CHAR, 1, 4) || '년' AS YEAR_,
       SUBSTR(DATE_CHAR, 5, 2) || '월' AS MONTH_,
       SUBSTR(DATE_CHAR, 7, 2) || '일' AS DATE_
FROM (
    SELECT '20210101' AS DATE_CHAR
    FROM DUAL
    UNION ALL
    SELECT '20220205' AS DATE_CHAR
    FROM DUAL
    UNION ALL
    SELECT '20230315' AS DATE_CHAR
    FROM DUAL
    UNION ALL
    SELECT '20240420' AS DATE_CHAR
    FROM DUAL
    UNION ALL
    SELECT '20250530' AS DATE_CHAR
    FROM DUAL
)
;
```

	YEAR_	MONTH_	DATE_
1	2021년	01월	01일
2	2022년	02월	05일
3	2023년	03월	15일
4	2024년	04월	20일
5	2025년	05월	30일

- SUBSTR함수는 문자단위로 시작위치와 길이를 지정하여 문자열을 자름
- SUBSTR(“문자열”, “시작위치”, “길이”)
- 참고로, SQL은 python과는 다르게 ‘1’부터 시작함

3. SUBSTR()

가상테이블

```
SELECT SUBSTR(DATE_CHAR, 1, 4) || '년' AS YEAR_,
       SUBSTR(DATE_CHAR, 5, 2) || '월' AS MONTH_,
       SUBSTR(DATE_CHAR, 7, 2) || '일' AS DATE_
FROM (
    SELECT '20210101' AS DATE_CHAR
    FROM DUAL
    UNION ALL
    SELECT '20220205' AS DATE_CHAR
    FROM DUAL
    UNION ALL
    SELECT '20230315' AS DATE_CHAR
    FROM DUAL
    UNION ALL
    SELECT '20240420' AS DATE_CHAR
    FROM DUAL
    UNION ALL
    SELECT '20250530' AS DATE_CHAR
    FROM DUAL
)
;
```

	YEAR_	MONTH_	DATE_
1	2021년	01월	01일
2	2022년	02월	05일
3	2023년	03월	15일
4	2024년	04월	20일
5	2025년	05월	30일

'MONTHS_컬럼'과 'DATE_'컬럼에 일의 자리 값이 입력되는 경우,
일의 자리 값만 표시되도록 하는 방법은?
(즉, '01월'을 '1월'로, '10월'은 '10월'로 표기하는 방법은?)

CASE & SUBSTR()

```
SELECT SUBSTR(DATE_CHAR, 1, 4) || '년' AS YEAR_,
       CASE
            WHEN SUBSTR(DATE_CHAR, 5, 1) = '0' THEN SUBSTR(DATE_CHAR, 6, 1)
            ELSE SUBSTR(DATE_CHAR, 5, 2)
       END || '월' AS MONTH_,
       CASE
            WHEN SUBSTR(DATE_CHAR, 7, 1) = '0' THEN SUBSTR(DATE_CHAR, 8, 1)
            ELSE SUBSTR(DATE_CHAR, 7, 2)
       END || '일' AS DATE_
FROM (
    SELECT '20210101' AS DATE_CHAR
    FROM DUAL
    UNION ALL
    SELECT '20220205' AS DATE_CHAR
    FROM DUAL
    UNION ALL
    SELECT '20230315' AS DATE_CHAR
    FROM DUAL
    UNION ALL
    SELECT '20240420' AS DATE_CHAR
    FROM DUAL
    UNION ALL
    SELECT '20250530' AS DATE_CHAR
    FROM DUAL
)
;
```

	YEAR_	MONTH_	DATE
1	2021년	1월	1일
2	2022년	2월	5일
3	2023년	3월	15일
4	2024년	4월	20일
5	2025년	5월	30일

JOIN 결과

```
SELECT A.SNO,
       A.SYEAR,
       A.SNAME,
       B.CNO,
       B.CNAME,
       C.RESULT
FROM SYSTEM.STUDENT A,
     SYSTEM.COURSE B,
     SYSTEM.SCORE C
WHERE A.SNO = C.SNO
      AND B.CNO = C.CNO;
```

	SNO	SYEAR	SNAME	CNO	CNAME	RESULT
58	943901	1	최혜원	1211	일반화학실험	58
59	943902	1	하정자	1211	일반화학실험	45
60	943903	1	유지아	1211	일반화학실험	35
61	948205	1	신형일	1211	일반화학실험	60
62	948209	2	유태지	1211	일반화학실험	77
63	894501	4	장봉혈	1212	일반화학	84
64	905301	4	유태지	1212	일반화학	87
65	905302	4	정욱상	1212	일반화학	82
66	905603	4	정용정	1212	일반화학	84
67	913901	4	황수현	1212	일반화학	97
68	913902	4	황진혜	1212	일반화학	70
69	913903	4	정도정	1212	일반화학	57

해당 결과에서, 학생별 과목 점수와 과목별 평균 점수 비교를 어떻게 할 수 있을까?

('과목별 평균 점수'컬럼 필요)

4. WINDOW

- WINDOW 함수는 '행과 행 간의 관계를 정의하기 위해서 제공되는 함수'임
- WINDOW 함수를 사용해서 순위, 합계, 평균, 행 위치 등을 조작할 수 있음
- PARTITION BY : 선택된 컬럼을 기준으로 전체 레코드 집합을 소그룹(파티션)단위로 묶음
  GROUP BY 구문과 비교했을 때, 둘 다 파티션을 분할한다는 의미에서 유사함
- ORDER BY : 선택된 컬럼을 기준으로 정렬
- **WINDOW 함수는 GROUP BY구문과 병행하여 사용할 수 없고, WINDOW 함수로 인해 결과 건수가 줄어들지 않음**

WINDOW 함수의 종류

구분	종류	종류
순위(RANK) 관련	RANK, DENSE_RANK, ROW_NUMBER	대부분 지원
집계(AGGREGATE) 관련	SUM, MAX, MIN, AVG, COUNT	SQL Server 경우 Over절 내 Orderby 지원 못함
순서 관련 함수	FIRST_VALUE, LAST_VALUE, LAG, LEAD	ORACLE 만 지원
그룹 내 비율 관련 함수	CUME_DIST, PERCENT_RANK, NTILE, RATIO_TO_REPORT	PERCENT_RANK 함수는 ANSI/ISO SQL 표준과 Oracle DBMS에서 지원하고 있으며, NTILE 함수는 ANSI/ISO SQL 표준에는 없지만, Oracle, SQL Server에서 지원하고 있다. RATIO_TO_REPORT 함수는 Oracle에서만 지원되는 함수(현업에서 유용).
선형분석을 포함한 통계분석 함수	CORR, COVAR_POP, COVAR_SAMP, STDDEV, STDDEV_POP, STDDEV_SAMP, VARIANCE, VAR_POP, VAR_SAMP, REGR_(LINEAR REGRESSION), REGR_SLOPE, REGR_INTERCEPT, REGR_COUNT, REGR_R2, REGR_AVGX, REGR_AVGY, REGR_SXX, REGR_SYY, REGR_SXY	특화되어있으므로 생략

WINDOW AVG()

```
SELECT A.SNO,
       A.SYEAR,
       A.SNAME,
       B.CNO,
       B.CNAME,
       C.RESULT,
       AVG(C.RESULT) OVER(PARTITION BY B.CNO) AS AVG_RESULT_BY_CNO
FROM SYSTEM.STUDENT A,
     SYSTEM.COURSE B,
     SYSTEM.SCORE C
WHERE A.SNO = C.SNO
      AND B.CNO = C.CNO;
```

	SNO	SYEAR	SNAME	CNO	CNAME	RESULT	AVG_RESULT_BY_CNO
58	943901	1	최혜원	1211	일반화학실험	58	68.37096774193548387096774193548387096774
59	943902	1	하정자	1211	일반화학실험	45	68.37096774193548387096774193548387096774
60	943903	1	유지아	1211	일반화학실험	35	68.37096774193548387096774193548387096774
61	948205	1	신형일	1211	일반화학실험	60	68.37096774193548387096774193548387096774
62	948209	2	유태지	1211	일반화학실험	77	68.37096774193548387096774193548387096774
63	894501	4	장봉철	1212	일반화학	84	69.13698630136986301369863013698630136986
64	905301	4	유태지	1212	일반화학	87	69.13698630136986301369863013698630136986
65	905302	4	정욱상	1212	일반화학	82	69.13698630136986301369863013698630136986
66	905603	4	정용정	1212	일반화학	84	69.13698630136986301369863013698630136986

- PARTITION BY에 'B.CNO'컬럼을 기입함으로써, 과목별 평균 점수를 산출할 수 있음

WINDOW AVG()

```
SELECT A.SNO,
       A.SYEAR,
       A.SNAME,
       B.CNO,
       B.CNAME,
       C.RESULT,
       AVG(C.RESULT) OVER(PARTITION BY B.CNO) AS AVG_RESULT_BY_CNO,
       CASE
            WHEN C.RESULT >= AVG(C.RESULT) OVER(PARTITION BY B.CNO) THEN '1'
            ELSE '0'
       END AS IS_OVER_AVG
FROM SYSTEM.STUDENT A,
     SYSTEM.COURSE B,
     SYSTEM.SCORE C
WHERE A.SNO = C.SNO
      AND B.CNO = C.CNO;
```

	SNO	SYEAR	SNAME	CNO	CNAME	RESULT	AVG_RESULT_BY_CNO	IS_OVER_AVG
1	894501	4	장봉철	1211	일반화학실험	97	68.37096774193548387096774193548387096774	1
2	905301	4	유태지	1211	일반화학실험	66	68.37096774193548387096774193548387096774	0
3	905302	4	정욱상	1211	일반화학실험	89	68.37096774193548387096774193548387096774	1
4	905603	4	정용정	1211	일반화학실험	81	68.37096774193548387096774193548387096774	1
5	913901	4	황수현	1211	일반화학실험	94	68.37096774193548387096774193548387096774	1
6	913902	4	황진혜	1211	일반화학실험	83	68.37096774193548387096774193548387096774	1

- WINDOW함수와 CASE문을 동시에 사용하면, 'IS_OVER_AVG'컬럼과 같은 BOOLEAN형 컬럼을 생성할 수 있음

JOIN 결과

```
SELECT A.SNO,
       A.SYEAR,
       A.SNAME,
       B.CNO,
       B.CNAME,
       C.RESULT
FROM SYSTEM.STUDENT A,
     SYSTEM.COURSE B,
     SYSTEM.SCORE C
WHERE A.SNO = C.SNO
      AND B.CNO = C.CNO;
```

	SNO	SYEAR	SNAME	CNO	CNAME	RESULT
58	943901	1	최혜원	1211	일반화학실험	58
59	943902	1	하정자	1211	일반화학실험	45
60	943903	1	유지아	1211	일반화학실험	35
61	948205	1	신형일	1211	일반화학실험	60
62	948209	2	유태지	1211	일반화학실험	77
63	894501	4	장봉혈	1212	일반화학	84
64	905301	4	유태지	1212	일반화학	87
65	905302	4	정욱상	1212	일반화학	82
66	905603	4	정용정	1212	일반화학	84
67	913901	4	황수현	1212	일반화학	97
68	913902	4	황진혜	1212	일반화학	70
69	913903	4	정도정	1212	일반화학	57

해당 결과에서, 과목별&학년별 점수 순위를 표현하는 방법은?

WINDOW RANK() & DENSE_RANK & ROW_NUMBER

```
SELECT SNAME,
       RESULT,
       RANK() OVER(ORDER BY RESULT DESC) AS RANK_FUNCTION,
       DENSE_RANK() OVER(ORDER BY RESULT DESC) AS DENSE_RANK_FUNCTION,
       ROW_NUMBER() OVER(ORDER BY RESULT DESC) AS ROW_NUMBER_FUNCTION
FROM (
    SELECT '오우재' AS SNAME,
           71 AS RESULT
    FROM DUAL
    UNION ALL
    SELECT '신형일' AS SNAME,
           80 AS RESULT
    FROM DUAL
    UNION ALL
    SELECT '최혜원' AS SNAME,
           92 AS RESULT
    FROM DUAL
    UNION ALL
    SELECT '유지아' AS SNAME,
           92 AS RESULT
    FROM DUAL
);
```

	SNAME	RESULT	RANK_FUNCTION	DENSE_RANK_FUNCTION	ROW_NUMBER_FUNCTION
1	최혜원	92	1	1	1
2	유지아	92	1	1	2
3	신형일	80	3	2	3
4	오우재	71	4	3	4

- RANK() : 중복 값들에 대해서 동일 순위로 표시하고, 중복 순위 다음 값에 대해서는 중복 개수만큼 떨어진 순위로 출력
- DENSE_RANK() : 중복 값들에 대해서 동일 순위로 표시하고, 중복 순위 다음 값에 대해서는 중복 값 개수와 상관없이 순차적인 순위 값을 출력
- ROW_NUMBER() : 중복 값들에 대해서도 순차적인 순위를 표시하도록 출력

WINDOW ROW_NUMBER()

```
SELECT A.SNO,
       A.SYEAR,
       A.SNAME,
       B.CNO,
       B.CNAME,
       C.RESULT,
       ROW_NUMBER() OVER(PARTITION BY B.CNO, SYEAR ORDER BY C.RESULT DESC) AS RNK_BY_CNO_N_SYEAR
FROM SYSTEM.STUDENT A,
     SYSTEM.COURSE B,
     SYSTEM.SCORE C
WHERE A.SNO = C.SNO
      AND B.CNO = C.CNO;
```

	SNO	SYEAR	SNAME	CNO	CNAME	RESULT	RNK_BY_CNO_N_SYEAR
1	925309	1	오우재	1211	일반화학실험	71	1
2	948205	1	신형일	1211	일반화학실험	60	2
3	943901	1	최혜원	1211	일반화학실험	58	3
4	943902	1	하정자	1211	일반화학실험	45	4
5	943903	1	유지아	1211	일반화학실험	35	5

- PARTITION BY에 'B.CNO'컬럼과 'SYEAR'컬럼을 기입함으로써, 과목별&학년별 점수 순위를 표현할 수 있음

JOIN 결과

```
SELECT A.SNO,
       A.SYEAR,
       A.SNAME,
       B.CNO,
       B.CNAME,
       C.RESULT
FROM SYSTEM.STUDENT A,
     SYSTEM.COURSE B,
     SYSTEM.SCORE C
WHERE A.SNO = C.SNO
      AND B.CNO = C.CNO;
```

	SNO	SYEAR	SNAME	CNO	CNAME	RESULT
58	943901	1	최혜원	1211	일반화학실험	58
59	943902	1	하정자	1211	일반화학실험	45
60	943903	1	유지아	1211	일반화학실험	35
61	948205	1	신형일	1211	일반화학실험	60
62	948209	2	유태지	1211	일반화학실험	77
63	894501	4	장봉혈	1212	일반화학	84
64	905301	4	유태지	1212	일반화학	87
65	905302	4	정욱상	1212	일반화학	82
66	905603	4	정용정	1212	일반화학	84
67	913901	4	황수현	1212	일반화학	97
68	913902	4	황진혜	1212	일반화학	70
69	913903	4	정도정	1212	일반화학	57

해당 결과에서, 학생별 과목 점수와 학년별&과목별 최대 점수·최소 점수를 어떻게 비교할 수 있을까?

('학년별&과목별 최대 점수'컬럼과 '학년별&과목별 최소 점수'컬럼 필요)

WINDOW FIRST_VALUE()

```
SELECT A.SNO,
       A.SYEAR,
       A.SNAME,
       B.CNO,
       B.CNAME,
       C.RESULT,
       FIRST_VALUE(C.RESULT) OVER(PARTITION BY B.CNO, A.SYEAR ORDER BY C.RESULT DESC) AS MAX_RESULT_BY_CNO_N_SYEAR,
       FIRST_VALUE(C.RESULT) OVER(PARTITION BY B.CNO, A.SYEAR ORDER BY C.RESULT) AS MIN_RESULT_BY_CNO_N_SYEAR,
       ABS(C.RESULT - FIRST_VALUE(C.RESULT) OVER(PARTITION BY B.CNO, A.SYEAR ORDER BY C.RESULT DESC)) AS COMPARED_WITH_MAX,
       ABS(C.RESULT - FIRST_VALUE(C.RESULT) OVER(PARTITION BY B.CNO, A.SYEAR ORDER BY C.RESULT)) AS COMPARED_WITH_MIN
FROM SYSTEM.STUDENT A,
     SYSTEM.COURSE B,
     SYSTEM.SCORE C
WHERE A.SNO = C.SNO
      AND B.CNO = C.CNO;
```

	SNO	SYEAR	SNAME	CNO	CNAME	RESULT	MAX_RESULT_BY_CNO_N_SYEAR	MIN_RESULT_BY_CNO_N_SYEAR	COMPARED_WITH_MAX	COMPARED_WITH_MIN
1	925309	1	오우재	1211	일반화학실험	71	71	35	0	36
2	948205	1	신형일	1211	일반화학실험	60	71	35	11	25
3	943901	1	최혜원	1211	일반화학실험	58	71	35	13	23
4	943902	1	하정자	1211	일반화학실험	45	71	35	26	10
5	943903	1	유지아	1211	일반화학실험	35	71	35	36	0
6	933904	2	임영현	1211	일반화학실험	90	90	45	0	45
7	933903	2	정승동	1211	일반화학실험	81	90	45	9	36

- FIRST_VALUE() 대신 WINDOW MAX()와 WINDOW MIN()을 사용해도 동일한 결과가 산출됨

가상테이블

```
SELECT 'A005930' AS SHRT_IS_CD,
       '20210415' AS BSNSS_DT,
       84100 AS CLS_PRC
FROM DUAL
UNION ALL
SELECT 'A005930' AS SHRT_IS_CD,
       '20210416' AS BSNSS_DT,
       83900 AS CLS_PRC
FROM DUAL
UNION ALL
SELECT 'A005930' AS SHRT_IS_CD,
       '20210419' AS BSNSS_DT,
       83300 AS CLS_PRC
FROM DUAL
UNION ALL
SELECT 'A005380' AS SHRT_IS_CD,
       '20210415' AS BSNSS_DT,
       230500 AS CLS_PRC
FROM DUAL
UNION ALL
SELECT 'A005380' AS SHRT_IS_CD,
       '20210416' AS BSNSS_DT,
       231500 AS CLS_PRC
FROM DUAL
UNION ALL
SELECT 'A005380' AS SHRT_IS_CD,
       '20210419' AS BSNSS_DT,
       230000 AS CLS_PRC
FROM DUAL
```

	SHRT_IS_CD	BSNSS_DT	CLS_PRC
1	A005930	20210415	84100
2	A005930	20210416	83900
3	A005930	20210419	83300
4	A005380	20210415	230500
5	A005380	20210416	231500
6	A005380	20210419	230000

해당 결과에 '전일 종가'컬럼과
'대비가(종가 – 전일 종가)'컬럼을 추가하려면?

LAG()

```
SELECT SHRT_IS_CD,
       BSNSS_DT,
       CLS_PRC,
       LAG(CLS_PRC) OVER(PARTITION BY SHRT_IS_CD ORDER BY BSNSS_DT) AS BFR_CLS_PRC,
       CLS_PRC - LAG(CLS_PRC) OVER(PARTITION BY SHRT_IS_CD ORDER BY BSNSS_DT) AS CHANGE_AMT
FROM (
    SELECT 'A005930' AS SHRT_IS_CD,
           '20210415' AS BSNSS_DT,
           84100 AS CLS_PRC
    FROM DUAL
    UNION ALL
    SELECT 'A005930' AS SHRT_IS_CD,
           '20210416' AS BSNSS_DT,
           83900 AS CLS_PRC
    FROM DUAL
    UNION ALL
    SELECT 'A005930' AS SHRT_IS_CD,
           '20210419' AS BSNSS_DT,
           83300 AS CLS_PRC
    FROM DUAL
    UNION ALL
    SELECT 'A005380' AS SHRT_IS_CD,
           '20210415' AS BSNSS_DT,
           230500 AS CLS_PRC
    FROM DUAL
    UNION ALL
    SELECT 'A005380' AS SHRT_IS_CD,
           '20210416' AS BSNSS_DT,
           231500 AS CLS_PRC
    FROM DUAL
    UNION ALL
    SELECT 'A005380' AS SHRT_IS_CD,
           '20210419' AS BSNSS_DT,
           230000 AS CLS_PRC
    FROM DUAL
);
```

	SHRT_IS_CD	BSNSS_DT	CLS_PRC	BFR_CLS_PRC	CHANGE_AMT
1	A005380	20210415	230500	(null)	(null)
2	A005380	20210416	231500	230500	1000
3	A005380	20210419	230000	231500	-1500
4	A005930	20210415	84100	(null)	(null)
5	A005930	20210416	83900	84100	-200
6	A005930	20210419	83300	83900	-600

- PARTITION BY에 'SHRT_IS_CD'컬럼을 기입하고 ORDER BY에 'BSNSS_DT'컬럼을 기입함으로써, 종목별 전일 종가 컬럼을 생성할 수 있음

PROFESSOR 테이블

```
SELECT *
FROM SYSTEM.PROFESSOR;
```

	PNO	PNAME	SECTION	ORDERS	HIREDATE
1	1001	송강	화학	정교수	02/08/12
2	1004	시진영	화학	부교수	91/02/01
3	1006	장청아	화학	부교수	03/05/20
4	1007	이초아	화학	조교수	10/07/06
5	1008	문규식	화학	조교수	05/02/11
6	1010	이규진	물리	정교수	98/10/07
7	1009	이준영	물리	정교수	91/10/04

해당 결과에 '여태 근무한 기간'을 나타내는 컬럼을 추가하려면?

SYSDATE

```
SELECT PNO,
       PNAME,
       HIREDATE,
       TRUNC(SYSDATE) - HIREDATE AS PERIODE
FROM SYSTEM.PROFESSOR;
```

	PNO	PNAME	HIREDATE	PERIODE
1	1001	송강	02/08/12	7001
2	1004	시진영	91/02/01	11211
3	1006	장청아	03/05/20	6720
4	1007	이초아	10/07/06	4116
5	1008	문규식	05/02/11	6087
6	1010	이규진	98/10/07	8406
7	1009	이준영	91/10/04	10966

- SYSDATE : 현재 시간(년, 월, 일, 시, 분, 초)를 표현하는 키워드
- 해당 키워드에 의해 생성되는 값들은 'DATE'타입
- 'TRUNC()'함수는 소수점 절사를 실시함
- 해당 'TRUNC()'함수를 SYSDATE에 적용하면, '시, 분, 초'는 절사됨
- DATE 타입 컬럼끼리 '-'연산이 가능하고, 해당 연산을 통해 두 일자 사이의 기간을 산출할 수 있음
- 'TRUNC(SYSDATE) – HIREDATE'를 통해, 여태 근무한 기간을 산출할 수 있음

SYSDATE

```
SELECT PNO,
       PNAME,
       HIREDATE,
       TRUNC(SYSDATE) - HIREDATE AS PERIODE
FROM SYSTEM.PROFESSOR;
```

	PNO	PNAME	HIREDATE	PERIODE
1	1001	송강	02/08/12	7001
2	1004	시진영	91/02/01	11211
3	1006	장청아	03/05/20	6720
4	1007	이초아	10/07/06	4116
5	1008	문규식	05/02/11	6087
6	1010	이규진	98/10/07	8406
7	1009	이준영	91/10/04	10966

해당 결과에서 'PERIODE'컬럼 기준 상위 3위 레코드만 조회하는 방법은?

SYSDATE

```
SELECT PNO,
       PNAME,
       HIREDATE,
       PERIODE,
       RNK
FROM (
    SELECT PNO,
           PNAME,
           HIREDATE,
           TRUNC(SYSDATE) - HIREDATE AS PERIODE,
           ROW_NUMBER() OVER(ORDER BY TRUNC(SYSDATE) - HIREDATE DESC) AS RNK
    FROM SYSTEM.PROFESSOR
)
WHERE RNK <= 3;
```

	PNO	PNAME	HIREDATE	PERIODE	RNK
1	1035	장관용	85/07/28	13225	1
2	1032	이유당	88/11/01	12033	2
3	1022	이준	89/05/05	11848	3

- PERIODE컬럼과 RNK컬럼을 내포한 쿼리 문을 INLINE VIEW로 구성
- 생성된 INLINE VIEW에 'RNK <= 3'조건문을 추가하여, 상위 3위 이내 레코드만 추출

6. ADD_MONTHS

PROFESSOR 테이블

```
SELECT PNO,
       PNAME,
       HIREDATE
FROM SYSTEM.PROFESSOR;
```

	PNO	PNAME	HIREDATE
1	1001	송강	02/08/12
2	1004	시진영	91/02/01
3	1006	장청아	03/05/20
4	1007	이초아	10/07/06
5	1008	문규식	05/02/11

'HIREDATE 컬럼'에 '+1달'또는 '-1달'을 적용 방법은?

6. ADD_MONTHS

ADD_MONTHS()

```
SELECT PNO,
       PNAME,
       HIREDATE,
       ADD_MONTHS(HIREDATE, 1) AS ONE_MONTH_PLUS,
       TO_CHAR(ADD_MONTHS(HIREDATE, 1), 'YYYY/MM/DD') AS ONE_MONTH_PLUS_WITH_FORMAT,
       ADD_MONTHS(HIREDATE, -1) AS ONE_MONTH_MINUS,
       TO_CHAR(ADD_MONTHS(HIREDATE, -1), 'YYYY/MM/DD') AS ONE_MONTH_MINUS_WITH_FORMAT
FROM SYSTEM.PROFESSOR;
```

	PNO	PN...	HIREDATE	ONE_MONTH_PLUS	ONE_MONTH_PLUS_WITH_FORMAT	ONE_MONTH_MINUS	ONE_MONTH_MINUS_WITH_FORMAT
1	1001	송강	02/08/12	02/09/12	2002/09/12	02/07/12	2002/07/12
2	1004	시진영	91/02/01	91/03/01	1991/03/01	91/01/01	1991/01/01
3	1006	장청아	03/05/20	03/06/20	2003/06/20	03/04/20	2003/04/20
4	1007	이초아	10/07/06	10/08/06	2010/08/06	10/06/06	2010/06/06
5	1008	문규식	05/02/11	05/03/11	2005/03/11	05/01/11	2005/01/11

- ADD_MONTHS()함수를 사용하면, 기존 DATE형식의 값에 입력한 정수만큼 N months가 더해짐
- ADD_MONTHS()함수를 적용한 결과의 형태를 변경시키고 싶다면, TO_CHAR()함수를 활용하여 해당 형태를 변경할 수 있음

7. LAST_DAY()

LAST_DAY()

```
SELECT TO_CHAR(HIREDATE, 'YYYY/MM/DD') AS STD_DT
FROM SYSTEM.PROFESSOR;
```

	STD_DT
1	2002/08/12
2	1991/02/01
3	2003/05/20
4	2010/07/06
5	2005/02/11
6	1998/10/07
7	1991/10/04
8	1999/04/19

'STD_DT'컬럼 기준, 해당 월 말일 컬럼과 전월 말일 컬럼을 산출하는 방법은?

LAST_DAY()

```
SELECT TO_CHAR(HIREDATE, 'YYYY/MM/DD') AS STD_DT,
       TO_CHAR(LAST_DAY(HIREDATE), 'YYYY/MM/DD') AS LAST_DT,
       TO_CHAR(TO_DATE(SUBSTR(TO_CHAR(HIREDATE, 'YYYYMMDD'), 1, 6) || '01') - 1, 'YYYY/MM/DD') AS BFR_MONTH_LAST_DT
FROM SYSTEM.PROFESSOR;
```

	STD_DT	LAST_DT	BFR_MONTH_LAST_DT
1	2002/08/12	2002/08/31	2002/07/31
2	1991/02/01	1991/02/28	1991/01/31
3	2003/05/20	2003/05/31	2003/04/30
4	2010/07/06	2010/07/31	2010/06/30
5	2005/02/11	2005/02/28	2005/01/31
6	1998/10/07	1998/10/31	1998/09/30
7	1991/10/04	1991/10/31	1991/09/30
8	1999/04/19	1999/04/30	1999/03/31
9	2000/05/18	2000/05/31	2000/04/30

- 월 말일 산출 : LAST_DAY()함수 사용(해당 함수는 입력되는 일자의 월 말일 값을 반환함)
- 월 초일 산출 : STD_DT의 '연월'까지 가져온 후, 해당 연월 뒤에 '01'을 붙여 월 초일을 산출함
  산출된 월 초일 값을 DATE형으로 변환한 뒤 '-1'연산을 수행하면, 전월 말일 값을 산출할 수 있음

8. 행 기반의 테이블을 열 기반의 테이블로 변환

JOIN 결과

```
SELECT A.SNAME,
       D.GRADE,
       COUNT(*) AS CNT
FROM SYSTEM.STUDENT A,
     SYSTEM.COURSE B,
     SYSTEM.SCORE C,
     SYSTEM.SCGRADE D
WHERE A.SNO = C.SNO
      AND B.CNO = C.CNO
      AND C.RESULT <= D.HISCORE
      AND C.RESULT >= D.LOSCORE
      AND A.SNO IN ('944503', '925602')
GROUP BY A.SNAME, D.GRADE
ORDER BY A.SNAME, D.GRADE;
```

	SNAME	GRADE	CNT
1	강아영	A	9
2	강아영	B	6
3	강아영	C	13
4	강아영	D	3
5	곽득용	A	5
6	곽득용	B	11
7	곽득용	C	9
8	곽득용	D	5
9	곽득용	F	1

SCGRADE 테이블

```
SELECT *
FROM SYSTEM.SCGRADE;
```

	GRADE	HISCORE	LOSCORE
1	A	100	85
2	B	84	70
3	C	69	55
4	D	54	40
5	F	39	0

1. 4개의 테이블을 조인 실시
2. STUDENT 테이블에서 SNO가 '944503', '925602'인 레코드를 필터링
3. 필터링 된 두 레코드와 SCORE 테이블을 INNER JOIN 실시
4. 직전 INNER JOIN결과와 COURSE 테이블을 INNER JOIN 실시
5. 직전 INNER JOIN결과와 SCGRADE 테이블을 JOIN 실시
6. 직전 JOIN 결과에서 SNAME컬럼과 GRADE컬럼으로 GROUP BY 실시
7. 조회된 '학생별&등급별 개수'결과를 SNAME컬럼과 GRADE컬럼 기준으로 ORDER BY 실시

8. 행 기반의 테이블을 열 기반의 테이블로 변환

JOIN 결과

```
SELECT A.SNAME,
       D.GRADE,
       COUNT(*) AS CNT
FROM SYSTEM.STUDENT A,
     SYSTEM.COURSE B,
     SYSTEM.SCORE C,
     SYSTEM.SCGRADE D
WHERE A.SNO = C.SNO
      AND B.CNO = C.CNO
      AND C.RESULT <= D.HISCORE
      AND C.RESULT >= D.LOSCORE
      AND A.SNO IN ('944503', '925602')
GROUP BY A.SNAME, D.GRADE
ORDER BY A.SNAME, D.GRADE;
```

	SNAME	GRADE	CNT
1	강아영	A	9
2	강아영	B	6
3	강아영	C	13
4	강아영	D	3
5	곽득용	A	5
6	곽득용	B	11
7	곽득용	C	9
8	곽득용	D	5
9	곽득용	F	1

해당 결과를 밑의 테이블 형태처럼 변경하는 방법은?

SNAME	A_CNT	B_CNT	C_CNT	D_CNT	F_CNT
강아영	9	6	13	3	0
곽득용	5	11	9	5	1

8. 행 기반의 테이블을 열 기반의 테이블로 변환

CASE 구문 활용

```
SELECT SNAME,
       CASE WHEN GRADE = 'A' THEN CNT ELSE 0 END AS A_CNT,
       CASE WHEN GRADE = 'B' THEN CNT ELSE 0 END AS B_CNT,
       CASE WHEN GRADE = 'C' THEN CNT ELSE 0 END AS C_CNT,
       CASE WHEN GRADE = 'D' THEN CNT ELSE 0 END AS D_CNT,
       CASE WHEN GRADE = 'F' THEN CNT ELSE 0 END AS F_CNT
FROM (
    SELECT A.SNAME,
           D.GRADE,
           COUNT(*) AS CNT
    FROM SYSTEM.STUDENT A,
         SYSTEM.COURSE B,
         SYSTEM.SCORE C,
         SYSTEM.SCGRADE D
    WHERE A.SNO = C.SNO
          AND B.CNO = C.CNO
          AND C.RESULT <= D.HISCORE
          AND C.RESULT >= D.LOSCORE
          AND A.SNO IN ('944503', '925602')
    GROUP BY A.SNAME, D.GRADE
    ORDER BY A.SNAME, D.GRADE
)
```

	SNAME	A_CNT	B_CNT	C_CNT	D_CNT	F_CNT
1	강아영	9	0	0	0	0
2	강아영	0	6	0	0	0
3	강아영	0	0	13	0	0
4	강아영	0	0	0	3	0
5	곽득용	5	0	0	0	0
6	곽득용	0	11	0	0	0
7	곽득용	0	0	9	0	0
8	곽득용	0	0	0	5	0
9	곽득용	0	0	0	0	1

- 직전 쿼리를 서브 쿼리형태(INLINE VIEW)로 감싼 후, 메인 쿼리에서 CASE 구문을 활용
- 해당 CASE 구문에 의해, 각 등급별 COUNT컬럼이 생성됨

8. 행 기반의 테이블을 열 기반의 테이블로 변환

최종 산출물

```
SELECT SNAME,
       SUM(A_CNT) AS A_CNT,
       SUM(B_CNT) AS B_CNT,
       SUM(C_CNT) AS C_CNT,
       SUM(D_CNT) AS D_CNT,
       SUM(F_CNT) AS F_CNT
FROM (
    SELECT SNAME,
           CASE WHEN GRADE = 'A' THEN CNT ELSE 0 END AS A_CNT,
           CASE WHEN GRADE = 'B' THEN CNT ELSE 0 END AS B_CNT,
           CASE WHEN GRADE = 'C' THEN CNT ELSE 0 END AS C_CNT,
           CASE WHEN GRADE = 'D' THEN CNT ELSE 0 END AS D_CNT,
           CASE WHEN GRADE = 'F' THEN CNT ELSE 0 END AS F_CNT
    FROM (
        SELECT A.SNAME,
               D.GRADE,
               COUNT(*) AS CNT
        FROM SYSTEM.STUDENT A,
             SYSTEM.COURSE B,
             SYSTEM.SCORE C,
             SYSTEM.SCGRADE D
        WHERE A.SNO = C.SNO
              AND B.CNO = C.CNO
              AND C.RESULT <= D.HISCORE
              AND C.RESULT >= D.LOSCORE
              AND A.SNO IN ('944503', '925602')
        GROUP BY A.SNAME, D.GRADE
        ORDER BY A.SNAME, D.GRADE
    )
)
GROUP BY SNAME;
```

	SNAME	A_CNT	B_CNT	C_CNT	D_CNT	F_CNT
1	곽득용	5	11	9	5	1
2	강아영	9	6	13	3	0

- 직전 쿼리를 서브 쿼리형태(INLINE VIEW)로 감싼 후, SNAME컬럼 기준 GROUP BY실시
- SELECT절에서 'SUM(등급별 카운트 컬럼)'을 각각 수행하면, 최종 결과가 산출됨
- 이처럼, 행 기반의 테이블을 열 기반의 테이블로 변환하면, 해당 테이블을 해석하기 쉬워 짐
- 쿼리를 통해서 최종 결과를 산출해야 할 때, 이 방법이 자주 활용됨