이 쿼리들은 `customer` 테이블을 대상으로 다양한 조건에 따라 데이터를 조회하고, 특정 조건에 맞게 데이터를 업데이트하는 작업을 수행합니다. 아래에 각 쿼리에 대한 설명을 제공합니다.

1. 고객 전체 검색

```
SELECT *
FROM customer;
```

설명:

- `customer` 테이블의 모든 열과 모든 행을 검색하여 반환합니다.

2. 고객 코드, 연락처, 담당자명, 회사명, 마일리지, 10% 추가된 마일리지 검색

```
SELECT cust_id
       ,contact_name
       ,company_name
       ,mileage AS points
       ,mileage * 1.1 AS "10%추가"
FROM customer;
```

설명:

- `cust_id`, `contact_name`, `company_name`, `mileage`를 선택하여 검색하며, `mileage`에서 10%가 추가된 값을 새로운 열로 계산해 표시합니다.

3. 마일리지가 100 이상인 고객 검색

```
SELECT cust_id
       ,contact_name
       ,mileage
FROM customer
WHERE mileage >= 100;
```

설명:

- `mileage`가 100 이상인 고객만 검색하여 `cust_id`, `contact_name`, `mileage`를 출력합니다.

4. 도시가 서울인 고객을 마일리지 역순으로 검색

```
SELECT cust_id
       ,contact_name
       ,city
       ,mileage AS points
FROM customer
WHERE city = '서울'
ORDER BY mileage DESC;
```

설명:

- `city`가 "서울"인 고객을 대상으로 `mileage`를 내림차순으로 정렬해 검색합니다.

5. 고객 3명만 검색

```
SELECT *
FROM customer
LIMIT 3;
```

설명:

- 테이블에서 처음 3명의 고객만 검색하여 반환합니다.

6. 마일리지 역순으로 정렬 후 3명 검색

```
SELECT *
FROM customer
ORDER BY mileage DESC
LIMIT 3;
```

설명:

- `mileage`를 내림차순으로 정렬한 후 상위 3명의 고객을 반환합니다.

7. 중복 없이 도시 이름만 검색

```
SELECT DISTINCT city
FROM customer;
```

설명:

- `city` 컬럼의 중복값을 제거하고 고유한 도시 이름만 반환합니다.

8. 산술 연산자 사용

```
SELECT 33 + 5 AS addition
      ,33 - 5 AS subtraction
      ,33 * 5 AS multiplication
      ,33 / 5::FLOAT AS float_division
      ,33 / 5 AS integer_division
      ,33 % 5 AS remainder1
      ,MOD(33, 5) AS remainder2;
```

설명:

- 다양한 산술 연산을 수행하고 그 결과를 각각의 열에 표시합니다.

9. 비교 연산자 사용

```
SELECT 33 >= 5 AS "greater_than_or_equal"
      ,33 <= 5 AS "less_than_or_equal"
      ,43 > 23 AS "greater_than"
      ,43 < 23 AS "less_than"
      ,43 = 43 AS "equal"
      ,43 != 23 AS "not_equal"
      ,43 <> 23 AS "not_equal_alternative";
```

설명:

- 다양한 비교 연산을 수행하고 그 결과를 반환합니다.

10. 직위가 대표가 아닌 고객 검색

```
SELECT *
FROM customer
WHERE contact_position <> '대표';
```

설명:

- `contact_position`이 "대표"가 아닌 고객만 검색하여 반환합니다.

11. 도시가 부산이고, 마일리지가 1000 이하인 고객 검색

```sql
SELECT *
FROM customer
WHERE city = '부산'
AND mileage < 1000;
```

설명:

- `city`가 "부산"이며 `mileage`가 1000 이하인 고객을 검색합니다.

12. 도시가 부산이거나 마일리지가 100 이하인 고객 검색

```sql
SELECT cust_id
       ,contact_name
       ,mileage
       ,city
FROM customer
WHERE city = '부산'
UNION
SELECT cust_id
       ,contact_name
       ,mileage
       ,city
FROM customer
WHERE mileage < 100
ORDER BY 1;
```

설명:

- 두 개의 쿼리 결과를 합쳐서 `city`가 "부산"이거나 `mileage`가 100 이하인 고객을 반환합니다.

13. 특정 고객의 지역을 NULL로 업데이트

```sql
UPDATE customer SET region = null WHERE cust_id = 4;
```

설명:

- `cust_id`가 4인 고객의 `region` 값을 `NULL`로 업데이트합니다.

14. 특정 고객의 지역을 빈 문자열로 업데이트

```sql
UPDATE customer SET region = '' WHERE cust_id = 13;
```

설명:

- `cust_id`가 13인 고객의 `region` 값을 빈 문자열(`''`)로 업데이트합니다.

15. 지역 데이터가 없는 고객 검색

```
SELECT *
FROM customer
WHERE region IS NULL;
```

설명:

- `region` 값이 `NULL`인 고객을 검색하여 반환합니다.

16. 지역이 빈 문자열인 고객 검색

```
SELECT *
FROM customer
WHERE region = '';
```

설명:

- `region` 값이 빈 문자열(`''`)인 고객을 검색하여 반환합니다.

17. 지역이 빈 문자열인 고객의 지역을 NULL로 업데이트

```
UPDATE customer
SET region = NULL
WHERE region = '';
```

설명:

- `region` 값이 빈 문자열인 고객의 `region`을 `NULL`로 업데이트합니다.

18. 직위가 대표이거나 사원인 고객 검색

```
SELECT cust_id
       ,contact_name
       ,contact_position
FROM customer
```

```
WHERE contact_position = '대표'
OR contact_position = '사원';
```

설명:

- `contact_position`이 "대표"이거나 "사원"인 고객을 검색하여 반환합니다.

19. 직위가 대표이거나 사원인 고객 검색 (IN 사용)

```
SELECT cust_id
       ,contact_name
       ,contact_position
FROM customer
WHERE contact_position IN ('사원', '대표');
```

설명:

- 위와 같은 결과를 `IN` 연산자를 사용하여 표현한 것입니다.

20. 마일리지가 100 이상 200 이하인 고객 검색

```
SELECT contact_name
       ,mileage
FROM customer
WHERE mileage >= 100
AND mileage <= 200;
```

설명:

- `mileage`가 100 이상 200 이하인 고객을 검색하여 반환합니다.

21. 마일리지가 100 이상 200 이하인 고객 검색 (BETWEEN 사용)

```
SELECT contact_name
       ,mileage
FROM customer
WHERE mileage BETWEEN 100 AND 200;
```

설명:

- `mileage`가 100 이상 200 이하인 고객을 `BETWEEN`을 사용하여 검색합니다.

22. 지역에 "광역시"가 포함된 고객 검색

```sql
SELECT *
FROM customer
WHERE region LIKE '%광역시';
```

설명:

- `region` 값에 "광역시"가 포함된 고객을 검색하여 반환합니다.

23. 서울에 거주하며, 마일리지가 150 이상 200 이하인 고객 검색

```sql
SELECT *
FROM customer
WHERE city LIKE '서울%'
AND mileage BETWEEN 150 AND 200;
```

설명:

- `city`가 "서울"로 시작하며 `mileage`가 150 이상 200 이하인 고객을 검색하여 반환합니다.

24. 중복 없는 지역과 도시 출력

```sql
SELECT DISTINCT region ,city
FROM customer
ORDER BY 1, 2;
```

설명:

- `region`과 `city`의 중복을 제거하고 고유한 값을 반환합니다.

25. 서울과 대전에 거주하며, 직위가 대표이거나 사원인 고객 검색

```sql
SELECT *
FROM customer
WHERE city IN ('서울', '대전')
AND (contact_position LIKE '%대표%' OR contact_position LIKE '%사원%');
```

설명:

- `city`가 "서울" 또는 "대전"이며 `contact_position`이 "대표"이거나 "사원"인 고객을 검색하여 반환합니다.

26. "특별시" 또는 "광역시"가 포함된 고객을 마일리지 역순으로 출력

```sql
SELECT *
FROM customer
WHERE NOT (city LIKE '%광역시' OR city LIKE '%특별시')
ORDER BY mileage DESC
LIMIT 3;
```

설명:

- `city`에 "광역시" 또는 "특별시"가 포함되지 않은 고객을 대상으로 `mileage` 역순으로 정렬한 후 상위 3명을 반환합니다.

27. 지역이 NULL이 아니고, 직위

가 대표가 아닌 고객 검색

```sql
SELECT *
FROM customer
WHERE region IS NOT NULL
AND contact_position <> '대표';
```

설명:

- `region` 값이 `NULL`이 아니며 `contact_position`이 "대표"가 아닌 고객을 검색하여 반환합니다.