セクション 10-ハンズオンスクリプト

■**等号**による論理式

まずは等号を利用した WEHRE 句の基本を確認してみましょう。

SELECT * FROM order_details WHERE product_id = 3;

と入力して、実行します。

当然ですが order_details テーブルから product_id が 3 番の注文情報のみが取得できます。まずはこれが基本の動きです。

そしてノット＝の動きを見ていきましょう。

SELECT * FROM order_details WHERE product_id != 3;

と入力して、実行します。

予想通り、order_details テーブルから product_id が 3 番以外の注文情報のみが取得できます。

つぎに

SELECT * FROM order_details WHERE product_id <> 3;

と入力して、実行してみましょう。

これは正確には不等号ですが、!=と同じ働きをします。

■不等号による論理式

今回は products テーブルを利用します。USE 文で指定します。

SELECT * FROM products;で products テーブルを確認すると

ID に製品名と価格が登録されていることがわかります。

この価格を不等号を利用した条件式を利用して、抽出する範囲を絞り込んでいきたいと思います。

まずは 40 ドルより上の価格となっている製品を抽出します。＞大なりを利用します。

SELECT * FROM products WHERE price > 40;

と入力して、実行します。

このリストには 40 ドルの製品も入っていたのですが、その製品は含まれずに、40 ドルを超過した製品のみが抽出されます。

これに＝を加えてみましょう。

SELECT * FROM products WHERE price >= 40;

すると、40 ドル以上の商品をリスト化します。つまり 40 ドルの製品も含まれています。

今度は不等号を逆にしてみましょう。

SELECT * FROM products WHERE price < 40;

で実行すると、40 ドルを含まない 40 未満の商品のみがリスト化されます。

これに等号を加えれば

SELECT * FROM products WHERE price <= 40;

当然に 40 ドルがリストに含まれるようになります。

そして、先ほどもやりましたが、

SELECT * FROM products WHERE price <> 40;

と入力することで、40 ドル以外のデータがすべて表示される！ =と同じ効果を得られます。

s

これで不等号の基本的な動作について確認できました。これは WHERE 句や HAVING 句

による条件付けの際に主に利用していくことになります。レクチャーを終了します。

■NOT LIKE による論理式

まずは LIKE 句の動作を確認してみましょう。

```
SELECT
    *
FROM
    users
WHERE
    birthday LIKE '20%';
```

と入力して、実行します。

これは users テーブルの中から、誕生日が 2000 年代のユーザーのみを抽出することができる条件文となります。つまり、%というワイルドカードを利用して、データの頭から 2 行が"20"となっていて、後ろはどんなデータであって、抽出しますよ！という条件を指定することができます。

先頭 2 桁が 20 になっています。

それではこの句を NOT LIKE にしていきましょう。

```
SELECT
    *
FROM
    users
WHERE
    birthday NOT LIKE '20%';
```

今度は 2000 年代以外の誕生日のユーザーのみが抽出されます。見てみると、先頭二桁が 19 代となっています。NOT LIKE を使うことで、LIKE 句によってワイルドカードによって広く指定されたデータを除外したデータのみに絞り込むことができるわけです。

■AND と OR による論理式

こちらは先ほどの LIKE 句の条件文です。

```
SELECT
    *
FROM
    users
WHERE
    birthday LIKE '20%';
```

実行すると、2000 年代の誕生日であるユーザーのみを抽出します。くわえて、性別による条件を加えて、男性のユーザーのみを抽出したいと考えます。

性別は gender というデータ項目に入力されています。それぞれ 0 が男性、1 が女性、2 がどちらでもない方を対象として分類しています。

```
SELECT
    *
FROM
    users
WHERE
    birthday LIKE '20%'
AND
    gender = 0;
```

AND を入力してから、追加の条件文を入力します。

これで実行します。

すると 2000 年代の誕生日であるユーザー＆男性であること。の 2 つの条件に合致したユーザーのみが抽出されます。

また AND を&&に変更することでも実行できます。これは別の書き方となります。

```
SELECT
	*
FROM
	users
WHERE
	birthday LIKE '20%'
&&
	gender = 0;
```

と AND から&&に変更して実行します。
すると先ほどと同じ結果が抽出されます。

これを OR 条件に変更してみましょう。

```
SELECT
	*
FROM
	users
WHERE
	birthday LIKE '20%'
OR
	gender = 0;
```

と AND から OR に変更して、実行します。

すると「2000 年代の誕生日であるユーザー」と「男性であること。」のどちらかの条件に合致しているユーザーが抽出されます。2 つの条件のどちらか一方がマッチしていれば良いため、先ほどよりも多くのユーザーが抽出されています。

■BETWEEN による論理式

```
SELECT
    *
FROM
    products
WHERE
    price
BETWEEN
    40 AND 598;
```

と入力して、実行します。

このように Products テーブルの中から、価格が 40 ドルから 598 ドルに収まっているデータが

全て抽出されます。

この効果は AND を利用した WEHER 句でも実現することができます。

```
SELECT
    *
FROM
    products
WHERE
    price >= 40
AND
    price <= 598;
```

と入力して、実行します。先ほどと同じ結果が抽出できたことがわかります。

このように BETWEEN 句の指定するデータは、>=　大なりイコールや小なりイコールといった、

＝も含めた条件となっているため、指定した数値も検索範囲に含まれているわけです。

次に先ほどの BETWEEN 句の方に NOT をつけて実行しています。

```
SELECT
	*
FROM
	products
WHERE
	price
NOT BETWEEN
	40 AND 598;
```

と入力して、実行します。このように、BETWEEN 句で指定した範囲外のデータが抽出されま

す。

これも WEHER 句にした場合にどういうクエリになるかを確認していきます。

```
SELECT
	*
FROM
	products
WHERE
	price < 40
OR
	price > 598;
```

条件式が AND ではなく、OR になることに注意してください。その上で、不等号が逆になって、＝がなくなります。これによって、NOT BETWEEN と同じ結果を絞り込むことができます。

■IN による論理式

今回は IN 句の後に 1 と 3 を指定します。これは、1~３までの範囲を取ってくるという意味で

す。

```
SELECT
	*
FROM
	order_details
WHERE
	product_id
IN (1, 3);
```

と入力して、実行します。

このように order_details テーブルの中から、Product_ID が１と３のデータを両方とも全て抽

出することができます。IN を利用することで、その中にあるデータを抽出するといった指定の仕

方ができるわけです。

また、IN 句を使用せずに OR を使うことで同じ結果を得ることもできます。

コマンドは下記の通りです。

```
SELECT
	*
FROM
	order_details
WHERE
		product_id = 1
```

```
    OR
    product_id = 3;
```

このように order_details テーブルの中から、Product_ID が１と３のデータを両方とも全て抽

出することができます。これは先ほどの IN(1,3)の結果と同じです。

指定範囲を変えてみましょう。

```
SELECT
    *
FROM
    order_details
WHERE
    product_id
IN (1);
```

１だけにすれば、Product_ID が１のデータが抽出されます

これは、

```
SELECT
    *
FROM
    order_details
WHERE
  product_id = 1;
```

と入力して、実行してみますと同じ結果となります。

さらに OR 条件をいれて、product_id = 3;を追加してます。

では、IN 句を使うメリットはなんでしょうか？

それは一言で言えば、省略可能であるということです。

つまり、IN を利用することで、条件を多数入力しなければならないケースにおいて、簡易に短いクエリコマンドで同じデータ抽出結果を実現することができるわけです。

たとえば、IN 句ならこのように書けばいいですが、

```
SELECT
	*
FROM
	order_details
WHERE
	product_id
IN (1,2,3,4,5);
```

と入力して、実行すると 5 つの Product_ID とマッチしたデータが抽出されます。

これを OR 句で実現してみましょう。

```
SELECT
	*
```

```
FROM
    order_details
WHERE
        product_id = 1
        OR
        product_id = 2
        OR
        product_id = 3
        OR
        product_id = 4
        OR
        product_id = 5;
```

と入力していきます。これで同じ結果が実行されますが、とても長くてめんどうなコマンドなのは

実感できたのではないでしょうか？

次に NOT を入れた場合の結果を確認していきます。このように BETWEEN などと同じよう

に、IN の前に NOT を指定することができます。

```
SELECT
    *
FROM
    order_details
WHERE
    product_id
NOT IN (1,2,3,4,5);
```

と入力して、実行しています。

これで、product_id が 1 ～ 5 以外のデータが抽出されます。これは IN (2,3,5,6,7,8,9);と実行した場合と同じとなります。

■ケーススタディ演習１解説

それでは演習１の回答を実施します。

まずは対象となるテーブルを選択して、データの中身を確認してみましょう。ユーザーを抽出するのですから、USERS テーブルからデータを抽出します。

SELECT * FROM users;

でテーブルの中身を見ていきましょう。

ここで名前を絞り込むことになるため、name 列を利用することになります。

頭文字の一部を利用してユーザーを抽出することになるため、LIKE 句を利用して、%を利用した検索を実施することが必要であることがわかります。

そして、それが ABC と３つ必要になるため、AND/OR による複数条件を並列させることが必要です。今回は頭文字 A または B または C となるため、OR を利用することが最適でしょう。

それでは入力してみます。

```
SELECT
    *
FROM
```

```
    users
WHERE
    name LIKE 'A%'
OR
    name LIKE 'B%'
OR
    name LIKE 'C%';
```

と入力して、実行しますと、頭文字 A または B または C となるユーザーのリストを抽出できました。

■ケーススタディ演習　2 解説

それでは演習 2 の回答を実施します。

今回は誕生日に基づいてユーザーを絞り込むことになるため、ユーザーテーブルの birthday 列を利用することになります。

SELECT * FROM users;

でテーブルの中身を見ていきます。

すると、birthday データは年月日が-で区分されたデータとして記録されているため、部分的に抽出するには、先ほどと同じように LIKE による部分検索を実施することが必要です。

そして年に 3 が含まれている場合と、月に 3 が含まれている場合の 2 つの条件がありますので、これも条件文としては OR となります。

しかしながら、実はこれは複数条件が必要となりません。

回答はこうなります。

```
SELECT
    *
FROM
    users
```

```
WHERE
    birthday LIKE '%3-%';
```

と入力して、実行することで年に３が含まれている場合と、月に３が含まれている場合の２つの条件を網羅して抽出することが可能です。

なぜ一文で出来たかというと、1978-03-12 というように、年と月の後ろには必ずハイフンがあるため、'%3-%'と指定してあげるだけで、年と月の両方を指定することができます。

複数条件はクエリの実行速度を落とすため、必要がない場合は極力短くすることが重要となりますので、今回の演習のように複数条件が必要ない要素をみつけるようにしてください。

■ケーススタディ演習　3 解説

それでは演習 3 の回答を実施します。

まずはどのように抽出すればよいのかを考えるため、全ての order_details テーブル内のデータを抽出してみます。

```
SELECT * FROM order_details;
```

今回はここから、order id と Product id に基づいてデータを絞り込むことが必要となります。

そのためには、まずは order id を LIKE を利用して 2 桁に絞り込む条件検索を実施します。

コマンドは下記の通りです。

```
SELECT
    *
FROM
    order_details
WHERE
    order_id LIKE '_';
```

と入力して、実行します。

これで order ID が 2 桁のデータのみが抽出されます。さらに product_id が 3,4,5,6 番の製品に対する発注のみにしぼりこみます。

```
SELECT
	*
FROM
	order_details
WHERE
	order_id LIKE '_'
AND
product_id
IN (3,4,5,6);
```

と入力して、実行します。

これで order ID と Product ID を複数指定してデータを絞り込むことができました。

■ケーススタディ演習　4 解説

それでは演習４の回答を実施します。

先ほどと同じようなクエリになりますので、直接にクエリを記述して回答していきたいと思います。

条件となるのは２つで、発注総量（amount）が２以上は amount >= 2 であり、プロダクト ID に対して NOT IN (5,6,7)を指定すれば製品 ID が５と６と９番以外の商品を指定することができます。

```
SELECT
　　*
FROM
　　order_details
WHERE amount >= 2
AND
product_id
NOT IN (5,6,7);
```

■ケーススタディ演習　5 解説

それでは演習 5 の回答を実施します。

このように同じカラムに対して、AND または OR を利用した複数条件を指定する場合で、数値を扱っている場合は BETWEEN 句に変更できるかを考えてください。

今回は

```
    price < 40
OR
    price > 300;
```

となっているところを BETWEEN 句で簡略化することができます。

まず変更前のコードを実行して、結果を確認してみましょう。

```
SELECT
    *
FROM
    products
WHERE
    price < 40
OR
    price > 300;
```

４０よりも低い価格または 300 よりも高い価格を条件として、６つの価格帯が表示されています。＝がないためその価格自体は含まれていません

これを BETWEEN 句にしますと

```
SELECT
    *
FROM
    products
WHERE
    price
NOT BETWEEN
    40 AND 300;
```

となります。これは 「40 <= PRICE <= 300」というように、40 ドル以上で 300 ドル以下となる

価格を除外したリストが抽出されることになり、先ほどの条件と同じになります。