さまざまな比較演算子

WHERE句などに使用される条件式には = や ＞ などの比較演算子を使って対象のデータの絞り込みを行いました。SQLには基本的な比較演算子以外にもさまざまな比較演算子がありますので今回はこちらを学習していきます。

# LIKE演算子

SQLでデータを絞り込む時に、ある値と同じ(=)や値以上(>)などの条件ではなく、

ある値を含むデータを絞り込む時にはLIKE演算子を使用します。LIKE演算子と以下のパターン文字を使うと部分一致検索が簡単に実現できます。

|  |  |
| --- | --- |
| パターン文字 | 意味 |
| % | 0文字以上の文字列 (文字が無くても良いし、複数文字でも良い) |
| \_ | 任意の1文字 |

ハンズオン　あいまい検索をする

１．商品表から商品名に「チキン」が含まれている商品データを取得する

SELECT \* FROM PRODUCT

WHERE PNAME LIKE '%チキン%' ;

|  |  |  |  |
| --- | --- | --- | --- |
| PRODUCT\_NO | PNAME | CATEGORY | PRICE |
| 0005  0006  1004 | チキン＆ベーコン  バーベキューチキン  チキンナゲット | ピザ  ピザ  サイド | 2200  2200  450 |

２．従業員表から１１月生まれの従業員名を取得する

SELECT ENAME , BIRTHDAY FROM EMPLOYEE

WHERE BIRTHDAY LIKE '\_\_\_\_-11-\_\_' ; --※\_が前に４つと後ろに2つ

|  |  |
| --- | --- |
| ENAME | BIRTHDAY |
| 河井ゆずる  丸山隆平  升野英知 | 1980-11-28  1983-11-26  1975-11-28 |

# BETWEEN演算子

BETWEEN演算子を使用すると範囲内のデータ取得が簡単に実装できます。例えば、価格が400円から600円の間の商品を取得する場合には以下のようにSQLを記述します。

WHERE PRICE BETWEEN 400 AND 600

なお、論理演算子を使うことで同様のSQL文を組み立てることも可能です。

WHERE PRICE >= 400 AND PRICE <= 600

ハンズオン　範囲内のデータを取得する

１．商品表から価格(PRICE)が400円以上、600円以下の商品情報を取得する

SELECT \* FROM PRODUCT

WHERE PRICE BETWEEN 400 AND 600 ;

# IN／NOT IN演算子

IN演算子はカッコの中の値に一致するデータの取得が出来ます。同じ列に対して複数の値を比較する時によく利用されます。

WHERE JOB IN ( '配達', '受付' )

なお、論理演算子を使うことで同様のSQL文を組み立てることも可能です。

WHERE JOB = '配達' OR JOB = '受付'

ハンズオン　リストに一致するデータを取得する

１．従業員表から職種(JOB)が配達か受付の従業員名を取得する

SELECT ENAME, JOB FROM EMPLOYEE

WHERE JOB IN ( '配達', '受付' ) ;

２．従業員表から職種(JOB)が配達か受付**以外**の従業員名を取得する

SELECT ENAME, JOB FROM EMPLOYEE

WHERE JOB NOT IN ( '配達', '受付' ) ;

# NULLの判定

データベースには「10」や「こんにちは」のような具体的な値ではなく、**NULL(ヌル)**というデータ不明の状態が存在します。実際には何もデータが格納されていないことを「NULLが格納されている」と表現されます。また、NULLはあくまで不明(未定義)なので**０とも空文字とも異なる**点に注意してください。

NULLとは

値が格納されていない、不明（未定義）の状態のこと

数字のゼロや長さ０文字の空文字とも異なる

そのため、NULLかどうかを判定するには=演算子などは使用できません。

何故なら＝は同じかどうかを判定する演算子だからです。

**NULL（不明）な値と同じですかと聞かれても、その答えはわからない（UNKNOWN）**となり条件式の判定が行えません。

列の値が、NULLであることを判定するには IS NULL演算子を使います。

つまりその値はNULLですか？と判定させる訳です。

NULL以外であることを判定するには IS NOT NULL演算子を使います。

ハンズオン　不明なデータを取得する

１．商品の原材料を管理する材料表(MATERIAL)のデータを表示する。

SELECT \* FROM MATERIAL ;

|  |  |  |  |
| --- | --- | --- | --- |
| MATERIAL\_NO | MNAME | ORIGIN | COST |
| 00001  00002  00003  …  00023 | チーズ  モッツァレラチーズ  パルメザンチーズ  …  メイプルソース | 北海道  北海道  イタリア  …  NULL | 50  700  450  …  30 |

２．産地(ORIGIN)が不明のデータを表示する。

SELECT \* FROM MATERIAL

WHERE ORIGIN IS NULL ;

ORDER BY 結果を並び替える

# 並び替えの基本

SELECT文によるデータ取得は表示順が順不同のため状況により検索結果が見にくくなってしまいます。検索結果を並び替えて表示したい場合は、SELECT文の最後にORDER BY句を記述して並び替えしたい列名を指定します。また、複数の列をカンマで区切ることで、並び替えの優先順位を設定することができます。

SELECT 列名 [, 列名] FROM 表名 WHERE 条件式

ORDER BY 列名 [, 列名]

※ORDER BY句の列名にDESCを付けると降順になる。

※MySQLは、並び替えを行う際に同じ値は順不同になる為、極力ユニークになるように列名を指定したほうが良い。

ハンズオン　並び替えをしてデータを表示する

１．商品情報を価格の安い順に並び変えてデータを取得する

SELECT \* FROM PRODUCT

ORDER BY PRICE ;

２．商品情報をカテゴリーごとの価格の高い順に並び変えてデータを取得する。

SELECT \* FROM PRODUCT

ORDER BY CATEGORY, PRICE DESC;

# LIMIT句について

ORDER BY句にLIMIT句を追加すると並び替えを行った結果の一部の行を取得することが出来ます。ランキングの上位５位や、10件ごとにデータを表示したいときに使用されます。

ORDER BY 列名 LIMIT 5 -- 先頭から５件のデータを表示する場合

ORDER BY 列名 LIMIT 5 OFFSET 10 -- 11件目から15件目を表示する場合

DISTINCT 重複行を除外する

SELECT句にDISTINCTキーワードを付加すると、列内で同じ値のデータを1つにまとめて表示することが出来ます。列名を指定したときに同じデータを表示したくないときなどに利用されます。

SELECT DISTINCT 列名

FROM 表名

ハンズオン　重複したデータを除外する

１．従業員表から役職情報を取得する

SELECT JOB FROM EMPLOYEE ;

２．重複したデータを除外して役職一覧を表示する

SELECT DISTINCT JOB FROM EMPLOYEE ;