**SQL(基本)**

**SQL(Structured Query Language)**

SQLは、関係データベースを定義・操作する言語で、データ定義言語(DDL)とデータ操作言語(DML)にわかれます。

データ定義言語は、データベースや表を作成するのに対して、データ操作言語は、表を使ってデータの抽出や挿入、更新、削除などを行います。

**SELECT文の基本形**

SELECTを使って、関係データベースの表から必要なデータを抽出します。必要なデータを抽出することを**照会、問い合わせ(クエリ)**ともいいます。

**基本形**

SELECT　列名1,列名2・・・抽出する列(射影すべき列)を指定する

FROM 表名1,表名2・・・抽出対象となる表を指定する

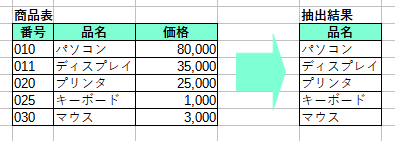
WHERE 条件式・・・ 抽出条件(選択すべき行の条件)を指定する

SELECT文は、「どの表から」「どういう条件で」「どの列を抽出するのか」が基本形となり、射影や選択、結合の関係演算を行うことができます。

## 射影

【例1】商品表から品名を抽出する

SELECT 品名 FROM 商品表;



## 選択

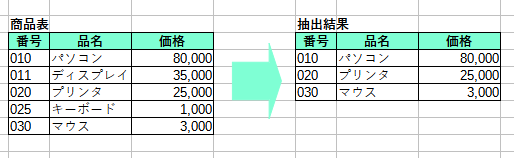
**WHERE句の抽出条件に、比較演算子や論理演算子を指定することができます。**

|  |  |  |
| --- | --- | --- |
|  | **構文** | **意味** |
| 比較演算子 | A = B | AはBと等しい |
| A <> B | AはBと等しくない |
| A > B | AはBより大きい |
| A < B | AはBより小さい(AはB未満) |
| A >= B | AはBと等しいか、より大きい(AはB以上) |
| A <= B | AはBと等しいか、より小さい(AはB以下) |
| 論理演算子 | A AND B | AかつB |
| A OR B | AまたはB |
| NOT A | Aではない |

【例2】商品表から番号が’010’,’020’,’030’のすべての列を抽出する

SELECT 番号,品名,価格 FROM 商品表

WHERE 番号 = ‘010’ OR 番号 = ‘020’ OR 番号 = ‘030’;



もっと簡単に書くのなら

【例2】で、ORの代わりに**IN**を使うことができます。不連続な値の数が多い場合に使います

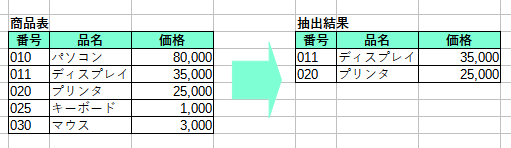
SELECT 番号,品名,価格 FROM 商品表

WHERE 番号 IN (‘010’,’020’,’030’);

【例3】商品表から、価格が1万円以上5万円以下のすべての列を抽出する。

SELECT 番号,品名,価格 FROM 商品表

WHERE 価格 >= 10000 AND 価格 <= 50000



もっと簡単に書くのなら

【例3】で、ANDの代わりに**BETWEEN**を使うことができます。上限と下限の範囲を指定します

SELECT 番号,品名,価格 FROM 商品表

WHERE 価格 BETWEEN 10000 AND 50000;

論理演算子を複数組合わせることができます。

その時の優先順位は、NOT、AND、ORの順に高く、さらに()を使って変更することができます。

## 結合

FROMで抽出対象となる表を複数指定し、WHERE句で表間をどの列で結合するかを指定します。

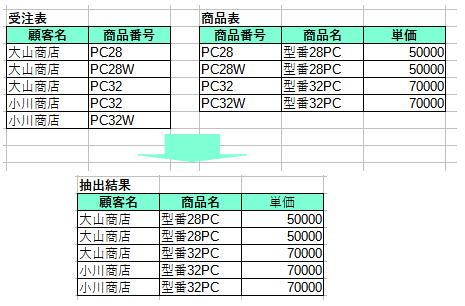
指定した複数の表に同じ列がある場合は、「表名.列名」で区別します。

【例4】受注表と商品表を結合し、顧客名と商品名、単価を抽出する。

SELECT 顧客名,商品名,単価

FROM 受注表,商品表

WHERE 受注表.商品番号 = 商品表.商品番号



重複を取り除く～DISTINCT～

DISTINCTを指定すると、重複した行を1つにまとめることができます

SELECT DISTINCT 顧客名,商品名,単価

FROM 受注表,商品表

WHERE 受注表.商品番号 = 商品表.商品番号

文字列部分一致～LIKE～

WHERE句にLIKEを使うと、文字列の一部分が一致するだけで抽出することができます。

|  |  |
| --- | --- |
| % | 0文字以上の任意の文字列 |
| \_ | 1文字の任意の文字列 |

社員表から、氏名に”三”の文字列を含む社員を抽出します。

SELECT \* FROM 社員表

WHERE 氏名 LIKE ‘%三’

【参考】

氏名 LIKE ‘三%’・・・氏名の最初が”三”で始まる社員

氏名 LIKE ‘%三’ ・・・氏名の最後が”三”で終わる社員

## 並べ替え

SELECT文で、**ORDER By**を指定すると、列の内容で昇順(ASC)または降順（DESC)に並べることができます。

なお、昇順に並べ替えを行う場合は、ASCを省略できます。

ORDER　BY 列名 ASC または DESC

【例1】日付の昇順に並べ替える

SELECT \* FROM 出庫記録

ORDER BY 日付 ASC

さらに、複数の列で並べ替えることもできます。

【例2】日付の昇順、さらに同じ日付の中で数量の降順で並べ替える。

SELECT \* FROM 出庫記録

ORDER BY 日付,数量 DESC

## 集合関係

指定した列の値を集計する集合関数(集約関数、集計関数)が用意されています。

|  |  |
| --- | --- |
| 関数 | 機能 |
| SUM (列名) | 指定した列の合計を求める　SELECT SUM(身長) FROM 社員表 |
| AVG (列名) | 指定した列の平均を求める SELECT AVG(身長) FROM 社員表 |
| MAX (列名) | 指定した列の最大値を求める SELECT MAX(身長) FROM 社員表 |
| MIN (列名) | 指定した列の最小値を求める SELECT MIN(身長) FROM 社員表 |
| COUNT (\*) | 指定した行数を求める SELECT COUNT(身長) FROM 社員表 |

## グループ化

SELECT文で**GROUP BY句**を指定すると、指定した列の内容が一致する行を１つの行にまとめる(グループ化する)ことができます。

GROUP BY 列名

【例3】販売から、商品コードごとの販売数量の合計を求め、商品コードと販売数量の合計を抽出する。

SELECT 商品コード,SUM(販売数量) FROM 販売表

GROUP BY 商品コード



そこで、SELECT直後の列名には、集合関数を除いてGROUP BY句に指定した列名以外を含めてはいけないことになっています。

## HAVING

さらに、GROUP BY句で指定したグループに対して、HAVINGを使って条件をつけることができます。

GROUP BY 列名 HAVING グループに対する条件

【列4】販売表から、商品コードごとの販売数量の合計が200を超える商品の商品コードと販売数量の合計を抽出する

SELECT 商品コード,SUM(販売数量) AS 合計数量 FROM 販売表

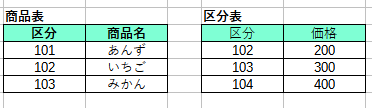
GROUP BY 商品コード HAVING SUM(販売数量) > 200

## 左部分結合

左表と右表を結合し、**左表のすべての行**を表示します。また、左表と対応する値が、右表にない場合、NULLを表示します。

SQLには、「LEFTY　OUTER JOIN」を記述します。

※OUTERは省略可能なため、LEFT　JOINでもよい。



【SQL】

SELECT　商品表.\*,区分表

FROM 商品表

LEFT OUTER JOIN 区分表 ON

商品表.区分 = 区分表.区分



なお、左外部結合の「左」とは、**JOIN**を中心に、その左側にある表という意味です。

### 結合条件の指定

左外部結合の結合条件は、ONに記述します。

【例】

SELECT 左表.\*, 右表.\*

FROM 左表 LEFT OUTER JOIN 右表

ONに結合条件式

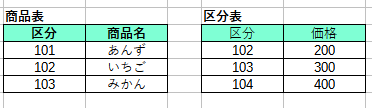
ON 左表.区分 = 右表.区分

## 右外部結合

左表と右表を結合し、右表のすべての行を表示します。また、右表と対応する値が、左表にない場合、NULLを表示します。

SQL文には「RIGHT OUTER JOIN」を記述します。

OUTERは省略可能なため、RIGHT JOINでもよい。



【SQL】

SELECT　商品表.\*,区分表

FROM 商品表

RIGHT OUTER JOIN 区分表 ON

商品表.区分 = 区分表.区分



104は、区分表は

あるが、商品表には

ないにNULLに。

### 結合条件の指定

右外部結合の結合条件は、ONに記述します。

【例】

SELECT 左表.\*, 右表.\*

FROM 左表 RIGHT OUTER JOIN 右表

ONに結合条件式

ON 左表.区分 = 右表.区分

## 副問合せ

かっこ内にあるSQL文のことで、かっこ外よりも前に実行します。副問合せ主問い合わせの違いは、次のとおりです。

主問い合わせ=()

外にあるSQL

【例】

副問い合わせ=()

内にあるSQL

SELECT \* FROM 社員表

WHERE 身長 <> ( SELECT MAX(身長) FROM 社員表 )

副問い合の実行順序

①副問合せを実行する、②その結果を使って、主問い合わせを実行する

## INSERT(データ挿入)

テーブルにデータを挿入する場合はINSERT文を使います。

【例】

INSERT INTO 商品表 (区分,商品名) VALUES(“104”,”ばなな”);

## UPDATE(データ更新)

テーブルにデータを更新する場合はUPDATEを使います

【例】

UPDATE 商品表

SET 商品名 = “パイナップル”

WHERE 区分 = “104”