2：データの操作(DML)入門4

製作：清水健二

SQLの中に別のSQLを書く

サブクエリ(副問い合わせ)

単純なSQLでは表示できないようなデータは、SQL文の中に別のSQL文を書くサブクエリを使用します。

WHERE句の中での副問い合わせ

ここではデータベース`world`を使用します。

```
USE world;
```

ex.世界中で人口が最も多い国の国名を1件表示する

下記のSQLでは世界の最大人口は表示できますが国名は表示できません。

```
SELECT MAX(Population) FROM Country;
```

そこで次に人口が「1277558000」人の国を絞り込んで表示します。

```
SELECT name,Population FROM Country WHERE Population =1277558000;
```

「中国」が表示されます。SQLを複数回実行しないとわからない結果については、下記のようにサブクエリを記述することで1回で結果を表示することができます。

```
SELECT Continent,name,Population
FROM Country
WHERE Population = (SELECT MAX(Population) FROM Country);
```

サブクエリの結果が複数件の場合の結果表示(IN句)

IN句は、対象データのいずれかと一致するものを取得できます。この対象データの抽出にサブクエリの結果を使うことができます。なお、抽出された条件に当てはまらないデータを抽出する場合は、NOT IN句を使います。
※NOT IN句を使うと、パフォーマンスが低下する場合があります。

ex.大陸ごとに人口が最も多い国の国名を複数件表示する
IN句を使うことで大陸ごとという複数件の検索結果に一致する国名を表示することができます。

```
SELECT Continent,name,Population
FROM Country
WHERE Population IN (SELECT MAX(Population) FROM Country GROUP BY Continent);
```

FROM句の中での副問い合わせ

テーブルの結合`JOIN`を行う際、予めサブクエリでデータを絞り込んでから結合を行うと処理が軽くなります。
ここでは、データベース`game_rank`を使用します。下記のコマンドを実行してください。

```
USE game_rank;
```

ex.サブクエリで売上が200万本以上のソフト名と本数を結合して表示する

```
SELECT sw.name,ss.sales_num
 FROM mt_software sw
 INNER JOIN
(
 SELECT *       ←software_id, sales_num
 FROM tt_software_sales
 WHERE sales_num>=2000000
) ss
ON sw.id = ss.software_id;
```

SELECT句の中での副問い合わせ

複数のテーブルのデータを表示したい場合にはJOINでの結合が便利ですが、SELECT句の中にサブクエリを書くことでフィールドに別テーブルの結果を表示することもできます。
テーブルの結合との違いは、集計関数も独立して別フィールドにできることです。

ex.サブクエリでゲームハード名、メーカー名、ランクインしているソフト数を表示する

```
SELECT md.name as ハード,
(
SELECT mb.name
 FROM mt_maker_brand mb
 WHERE mb.id = md.maker_brand_id
) as メーカー,
(
SELECT COUNT(sw.id)
 FROM mt_software sw
 WHERE sw.models_cd = md.cd
) as ランクイン数
 FROM mt_models md;
```

ex.サブクエリで横並びにデータを表示する

EXCELで見かけるようなクロス集計表形式のテーブルを表示したい場合もサブクエリを使えば実現できます。

```
SELECT DISTINCT
(
SELECT COUNT(fc.models_cd)
 FROM mt_software fc   //ソフト
 WHERE fc.models_cd = 'FC'   // ハード
) as FC,
(
SELECT COUNT(sfc.models_cd)
 FROM mt_software sfc
 WHERE sfc.models_cd = 'SFC'
) as SFC,
(
SELECT COUNT(ps.models_cd)
 FROM mt_software ps
 WHERE ps.models_cd = 'PS'
) as PS
 FROM mt_software sw;
```

INSERT,UPDATE文でサブクエリを使用する

サブクエリはSELECT文だけでなく、INSERT、UPDATE文でも利用できます。例えば、別テーブルで用意した2015年以降のゲームランキングデータを検索して、その結果を今までのランキングデータの続きに挿入したり、WHERE句で指定したサブクエリの結果に一致したものだけ更新の対象にする等、使える幅は広いです。サブクエリを使いこなすには慣れが必要ですが、難解な問題に対して複雑なプログラムを書く前に、SQLやサブクエリで解決できないか考えてみるとよいでしょう。

データの存在を確認する

サブクエリの結果が存在する場合のみ検索結果を表示(EXISTS)

サブクエリの実行結果でデータが存在しない場合、メインのSQLも終了してしまいます。
そこで、検索結果が返ってくるデータのみ表示できる`EXISTS`句を使用します。
ここでは`world`のデータベースを使用します。 下記のコマンドを実行してください。

```
USE world;
```

ex.都市人口が5000万人以上の都市を持つ国名を表示する

```
SELECT cn.Name
 FROM Country cn
 WHERE EXISTS
(
 SELECT *
 FROM City ci
 WHERE ci.Population >=5000000
 AND ci.CountryCode = cn.code
);
```

別のテーブルのひもづけに必要？？

city

country

参考

https://www.sejuku.net/blog/73615

EXISTS句が無い場合、メインクエリの方に5000万人未満の都市も検索結果に含まれるためエラーになりますが、EXISTS句をつけることで、5000万人以上の都市が存在しているデータのみ表示されます。

複数のテーブルの結果を１つにまとめる

複数のテーブルを結果を１つのテーブルにまとめる(UNION)

ひとまず、下記のSQLを実行してみてください。

```
SELECT cn.Name,cn.Population
 FROM Country cn
 WHERE cn.Code = 'JPN'
 UNION
 SELECT ci.Name,ci.Population
 FROM City ci
 WHERE ci.CountryCode = 'JPN';
```

国データ日本の国名と人口のテーブルと、日本の都市データの都市名と人口が入ったテーブルがまるで一つのテーブルのように表示されます。UNIONは結合するそれぞれのクエリーのカラムの数、順番、データ型が一致していれば、たとえ別のテーブルであっても縦方向に結合することが可能です。

よく使う検索結果をテーブルのように扱う

検索結果をテーブルとして扱う(VIEW)

ここでは、データベース`game_rank`を使用します。下記のコマンドを実行してください。

```
USE game_rank;
```

ex.ドラゴンクエストシリーズの売上表をVIEWにする

下記のコマンドを実行してください。

```
SELECT sw.name,sw.sale_at,sw.models_cd,ss.sales_num
FROM mt_software sw
INNER JOIN tt_software_sales ss
ON sw.id = ss.software_id
WHERE sw.name LIKE 'ドラゴンクエスト%'
ORDER BY sale_at;
```

毎回データを確認するのにこのようなSQLを打たなければならないのは面倒なことです。
そこで、よく使うテーブルの検索結果をVIEW機能を使って保存します。
INNER JOINなどを何回もしないといけないような場面では、あらかじめINNER JOINしたビューを作成しておくと便利に使えます。

VIEWの作成(CREATE VIEW)

ここでは新規のVIEW `v_dq`を作成します。

```
CREATE VIEW v_dq
(
  シリーズ名,発売日,ハード,販売数
) AS
SELECT sw.name,sw.sale_at,sw.models_cd,ss.sales_num
FROM mt_software sw
INNER JOIN tt_software_sales ss          //内部結合（ソフトウェアテーブルと売り上げテーブル）
ON sw.id = ss.software_id                //ひもづけ
WHERE sw.name LIKE 'ドラゴンクエスト%'
ORDER BY sale_at;                        売り上げ数昇順
```

VIEWの確認

VIEWが作成できたか`SHOW TABLES`で確認できます。

```
SHOW TABLES;
```

VIEWの使用

VIEWは通常のテーブルのように使用できます。

```
SELECT * FROM v_dq;
```

VIEWは内容が追加変更されても反映されます。しかし、実行速度は遅いため、VIEW同士をJOINするなど負荷の大きな処理には多用しないでください。