

de-13. データ管理の基礎：オンラインランザクシヨンとデータウェアハウスの特徴と活用

(データベース演習)

URL: <https://www.kkaneko.jp/de/de/index.html>

金子邦彦



データの一元管理と保存の利点

• データの一元管理

さまざまな種類のデータを一か所で管理することで、データアクセスと利用が容易になる。

• データの長期保存

長期間にわたるデータの安全な保存が容易になる

• データ分析

さまざまな種類のデータを組み合わせた分析が可能になる

データウェアハウスとオンライントランザクションの比較

- ・ **オンライントランザクション** : **最新情報のみを使用**

予約テーブル

氏名	予約内容
XX	おせちA
YY	おせちB
ZZ	おせちA

価格テーブル

商品	価格
おせちA	10000
おせちB	5000

- ・ **データウェアハウス** : **「履歴データ」を重視する**

予約テーブル

氏名	予約内容	予約日	キャンセル日
XX	おせちA	2023-12-01	
YY	おせちB	2023-12-04	
ZZ	おせちB	2023-12-05	2023-12-06
ZZ	おせちA	2023-12-06	

価格テーブル

商品	価格	価格改定日
おせちA	12000	2023-11-10
おせちA	10000	2023-11-20
おせちB	5000	2023-11-10

履歴データの概念

- データの各行に日時情報を付加
- データ変化があるたびに新しい行を挿入
- 一度保存されたデータは削除しない
- 過去のデータの変化の履歴を保持

予約テーブル

氏名	予約内容	予約日	キャンセル日
XX	おせちA	2023-12-01	
YY	おせちB	2023-12-04	
ZZ	おせちB	2023-12-05	2023-12-06
ZZ	おせちA	2023-12-06	

価格テーブル

商品	価格	価格改定日
おせちA	12000	2023-11-10
おせちA	10000	2023-11-20
おせちB	5000	2023-11-10

「商品の価格は1つに決まっている」という考え方ではなく、
商品の価格の履歴データを保持

データの一元管理と保存

2つの異なるアプローチ

	データウェアハウス	オンライントランザクション
主な目的	データ分析、データからのルール発見	銀行の取引、オンライン予約、販売管理
機能	長期間のデータを用いたデータ分析	トランザクション処理
使用される技術	SQL、データの挿入、結合、グループ化	SQL、データの挿入と削除と更新、単純な問い合わせ
主な理念	履歴データ	正規化

データ管理のまとめ

データウェアハウス

- 「履歴データ」の利用
- データの時間の時間による分析を可能にする
- **一度保存されたデータは恒久的に保持**される
- 削除や変更は原則として行われない

オンライントランザクション

- 最新のデータを保持する
- データの値の変更や削除に伴い異状が発生する可能性がある。
- 正規化によりデータの冗長性を排除し、整合性を確保する

Access での注意点

- SQLビューでは、SQL文を1つずつ実行
(複数まとめての一括実行ができない)
- **CREATE TABLE** では、「実行」の後、**画面が変化しない**が実行できている
- **INSERT INTO** では、「実行」の後、**確認表示**が出る。その後、**画面が変化しない**が実行できている

Access でのテーブルデータの確認

- SQL で確認

```
SELECT * FROM T;
```

	名前	昼食	料金
A		そば	250
B		カレーライス	400
C		カレーライス	400
D		うどん	250
*			

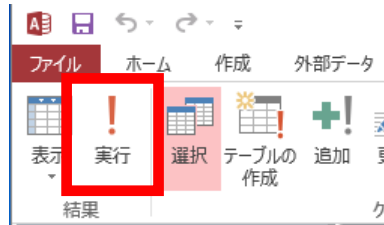
- テーブルビューで、「テーブル名」をダブルクリック

The screenshot shows the Microsoft Access interface. The ribbon is set to 'ホーム' (Home) under the 'テーブルの' (Tables) group. The ribbon includes icons for '表示' (View), '貼り付け' (Paste), 'フィルター' (Filter), '並べ替えとフィルター' (Sort & Filter), 'すべて更新' (Refresh All), 'レコード' (Records), and '検索' (Search). Below the ribbon, the 'すべて...' (All) pane shows a search box and a list of tables containing 'T'. The main window displays a table named 'T' with the following data:

	名前	昼食	料金
A		そば	250
B		カレーライス	400
C		カレーライス	400
D		うどん	250
*			

SQL 問い合わせ（クエリ）で使用する2つのビュー

```
SELECT * from 商品;
```



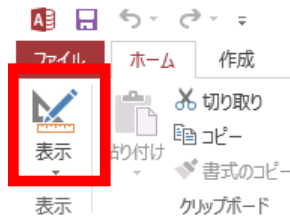
実行



ID	名前	単価
	みかん	50
	2りんご	100
	3りんご	150
*	(新規)	0

SQL ビュー

SQL 文の 作成、編集



表示 + SQL ビュー



データシートビュー

問い合わせ（クエリ）の
結果

マウス操作でビューを切り替え

Access の SQL ビューを用いた問い合わせ

① Access の **SQLビュー**開く

② **SQL 文**の編集。 **select, from, where** を使用

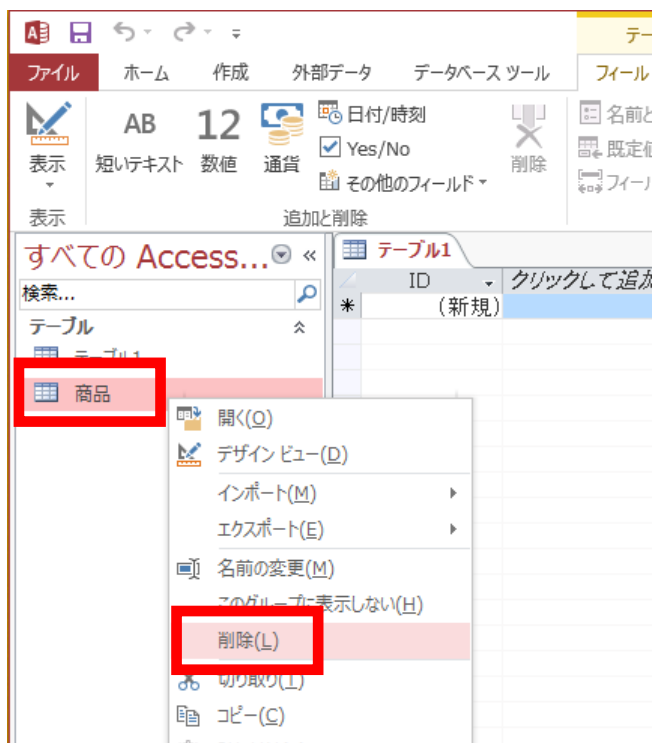
例: `select * from テーブル名 where 列1 = 値1;`

③ **SQL 文**の**実行**

実行の結果、**データシートビュー**に画面が変わり、そこに**問い合わせの結果**が表示される

④ さらにSQL 文の編集、実行を続ける場合には、**画面を SQL ビューに切り替える**

間違ってしまったときは、テーブルの削除 を行ってからやり直した方が早い場合がある



テーブルビューで、削除したいテーブルを**右クリック**して、**「削除」**

テーブルを削除するときは、
間違っても必要な**テーブル**を削除しない
ように、十分に注意する！
(元に戻せない)

演習の手順

1. パソコンを使用する

前もって Access をインストールしておくこと

2. Access を起動する

3. Access で、「**空のデータベース**」を選び、「**作成**」をクリック。

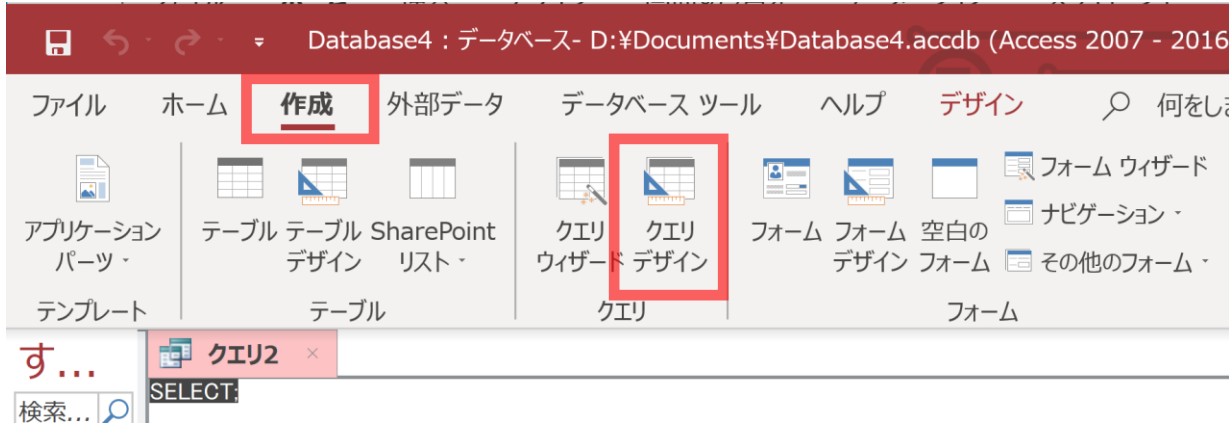


4. テーブルツール画面が表示されることを確認

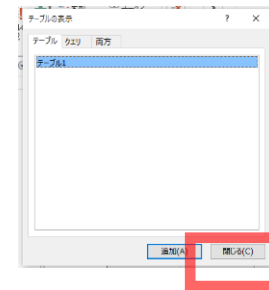
The screenshot displays the Microsoft Access 2016 interface. The title bar shows the file name "Database7 : データベース- D:\¥Documents¥Database7.accdb (Access 2007 - 2016 ファイル形式)..." and the user name "金子 邦彦". The ribbon is set to "フィールド" (Fields) under the "テーブル" (Table) group. The ribbon includes sections for "名前と標題" (Name and Title), "既定値" (Default Value), "フィールド サイズ" (Field Size), "プロパティ" (Properties), "ルックアップの変更" (Change Lookup), "fx 式の変更" (Change Formula), "メモの設定" (Memo Settings), "書式設定" (Formatting), "表示形式" (Display Format), and "フィールドの入力規則" (Field Validation Rules). The main area shows a table named "テーブル1" with a single column "ID" and one row containing "(新規)". A yellow highlight is on the "ID" header, and a blue highlight is on the "(新規)" cell. The status bar at the bottom indicates "レコード: 1 / 1" and "フィルターなし 検索".

ID
(新規)

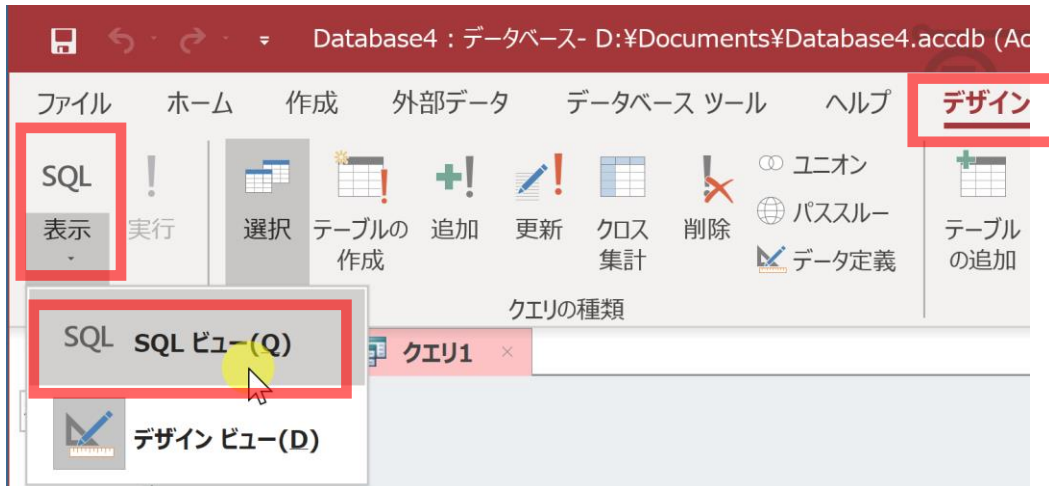
5. 次の手順で、SQLビューを開く。



① 「作成」タブで、「クエリデザイン」をクリック



このような表示が出たときは「閉じる」をクリック



② 「デザイン」タブで、「表示」を展開し「SQLビュー」を選ぶ



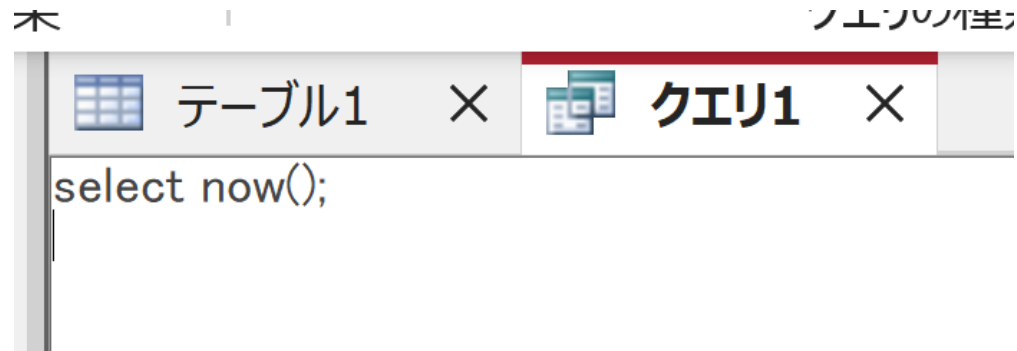
演習 1. `now()` による現在 日時の取得

【トピックス】

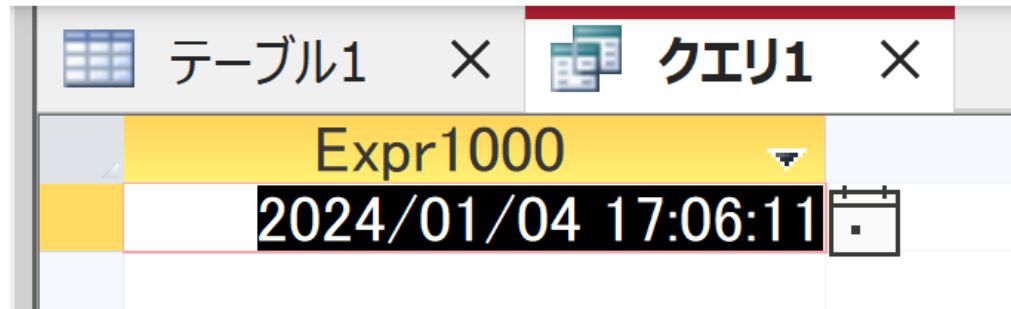
1. `now()`

SQL ビューに、次の SQL を入れ、「実行」ボタンで、SQL 文を実行。結果を確認

```
select now();
```



実行結果



A screenshot of the SQL editor showing the execution result. The top bar shows the same tabs as the previous screenshot. Below the query editor, a table is displayed with one row. The first column is labeled 'Expr1000' and contains the value '2024/01/04 17:06:11'. The table has a yellow header row and a black background for the data row.

Expr1000
2024/01/04 17:06:11

表示の幅はマウス操作で
変えることができる



演習 2. 日時を扱うテーブル

【トピックス】

1. 日時
2. DATETIME

① **SQL ビュー**に、次の SQL を1つずつ入れ、「**実行**」ボタンで、**SQL文**を実行。結果を確認

```
CREATE TABLE 商品 (  
    ID INTEGER PRIMARY KEY,  
    商品名 TEXT,  
    単価 INTEGER);  
  
CREATE TABLE 購入 (  
    ID INTEGER PRIMARY KEY,  
    購入者 TEXT,  
    商品ID INTEGER,  
    数量 INTEGER,  
    購入日時 DATETIME,  
    FOREIGN KEY (商品ID) REFERENCES 商品(ID));
```

② **SQL ビュー**に、次の SQL を1つずつ入れ、「**実行**」ボタンで、**SQL文**を実行。結果を確認

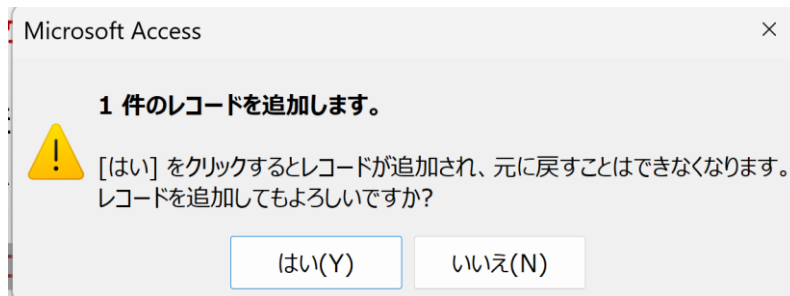
```
INSERT INTO 商品 VALUES (1, 'みかん', 50);
```

```
INSERT INTO 商品 VALUES (2, 'りんご', 100);
```

```
INSERT INTO 商品 VALUES (3, 'メロン', 500);
```

```
INSERT INTO 購入 VALUES (1, 'X', 1, 10, '2023-01-04 09:00:00');
```

```
INSERT INTO 購入 VALUES (2, 'Y', 2, 5, '2023-01-04 10:00:00');
```



INSERT INTOでは、「実行」の後、確認表示が出る。その後、**画面が変化しない**が実行できている

③ SQL ビューに、次の SQL を1つずつ入れ、「実行」ボタンで、SQL文を実行。結果を確認

```
select * FROM 商品;
```

実行結果

ID	商品名	単価
1	みかん	50
2	りんご	100
3	メロン	500

④ SQL ビューに、次の SQL を1つずつ入れ、「実行」ボタンで、SQL文を実行。結果を確認

```
select * FROM 購入;
```

実行結果

ID	購入者	商品ID	数量	購入日時
1X		1	10	2023/01/04 9:00:00
2Y		2	5	2023/01/04 10:00:00



演習 3. データウェアは渦 の構築

【トピックス】

1. 日時
2. DATETIME
3. 履歴データ

① **SQL ビュー**に、次の SQL を1つずつ入れ、「**実行**」ボタンで、**SQL文**を実行。結果を確認

```
CREATE TABLE 商品 (  
    ID INTEGER PRIMARY KEY,  
    商品名 TEXT,  
    単価 INTEGER,  
    改訂日時 DATETIME);
```

```
CREATE TABLE 購入 (  
    ID INTEGER PRIMARY KEY,  
    購入者 TEXT,  
    商品ID INTEGER,  
    数量 INTEGER,  
    購入日時 DATETIME,  
    FOREIGN KEY (商品ID) REFERENCES 商品 (ID));
```

② SQL ビューに、次の SQL を1つずつ入れ、「実行」ボタンで、SQL文を実行。結果を確認

```
INSERT INTO 商品 VALUES (1, 'みかん', 50, '2023-12-01 09:00:00');
```

```
INSERT INTO 商品 VALUES (2, 'りんご', 100, '2023-12-01 09:00:00');
```

```
INSERT INTO 商品 VALUES (3, 'メロン', 500, '2023-12-01 09:00:00');
```

```
INSERT INTO 商品 VALUES (4, 'りんご', 150, '2024-01-01 09:00:00');
```

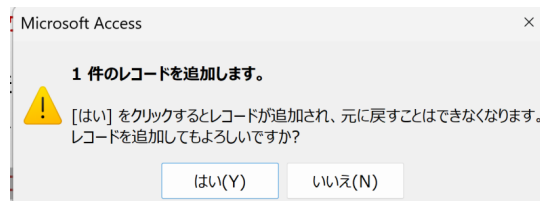
```
INSERT INTO 商品 VALUES (5, 'メロン', 400, '2024-01-01 09:00:00');
```

```
INSERT INTO 購入 VALUES (1, 'X', 1, 10, '2023-12-10 10:00:00');
```

```
INSERT INTO 購入 VALUES (2, 'Y', 2, 5, '2023-12-20 12:00:00');
```

```
INSERT INTO 購入 VALUES (3, 'Y', 4, 20, '2024-01-05 09:00:00');
```

```
INSERT INTO 購入 VALUES (4, 'Z', 5, 3, '2024-01-05 11:00:00');
```




INSERT INTOでは、「実行」の後、確認表示が出る。その後、画面が変化しないが実行できている

③ SQL ビューに、次の SQL を入れ、「実行」ボタンで、SQL文を実行。結果を確認

```
select * FROM 商品;
```

実行結果



The screenshot shows a window titled 'クエリ1' (Query 1) displaying the results of a SQL query. The table has 5 columns: ID, 商品名 (Product Name), 単価 (Unit Price), and 改訂日時 (Revision Date/Time). The first row is highlighted in blue. The asterisk (*) in the bottom-left corner indicates that the table is scrollable.

ID	商品名	単価	改訂日時
1	みかん	50	2023/12/01 9:00:00
2	りんご	100	2023/12/01 9:00:00
3	メロン	500	2023/12/01 9:00:00
4	りんご	150	2024/01/01 9:00:00
5	メロン	400	2024/01/01 9:00:00

表示の幅はマウス操作で
変えることができる

④ SQL ビューに、次の SQL を入れ、「実行」ボタンで、SQL文を実行。結果を確認

```
select * FROM 購入;
```

実行結果

ID	購入者	商品ID	数量	購入日時
1X		1	10	2023/12/10 10:00:00
2Y		2	5	2023/12/20 12:00:00
3Y		4	20	2024/01/05 9:00:00
4Z		5	3	2024/01/05 11:00:00
*				

⑤ **SQLビュー**に、次のSQLを1つずつ入れ、「**実行**」ボタンで、**SQL文**を実行。結果を確認

結合

```
SELECT * FROM 商品  
INNER JOIN 購入 ON 商品.ID = 購入.商品ID;
```

```
SELECT 購入.購入日時, 購入.購入者, 購入.数量 * 商品.単価  
FROM 商品  
INNER JOIN 購入 ON 商品.ID = 購入.商品ID;
```

商品.ID	商品名	単価	改訂日時	購入.ID	購入者	商品ID	数量
1	みかん	50	2023/12/01 9:00:00	1X		1	10
2	りんご	100	2023/12/01 9:00:00	2Y		2	5
4	りんご	150	2024/01/01 9:00:00	3Y		4	20
5	メロン	400	2024/01/01 9:00:00	4Z		5	3

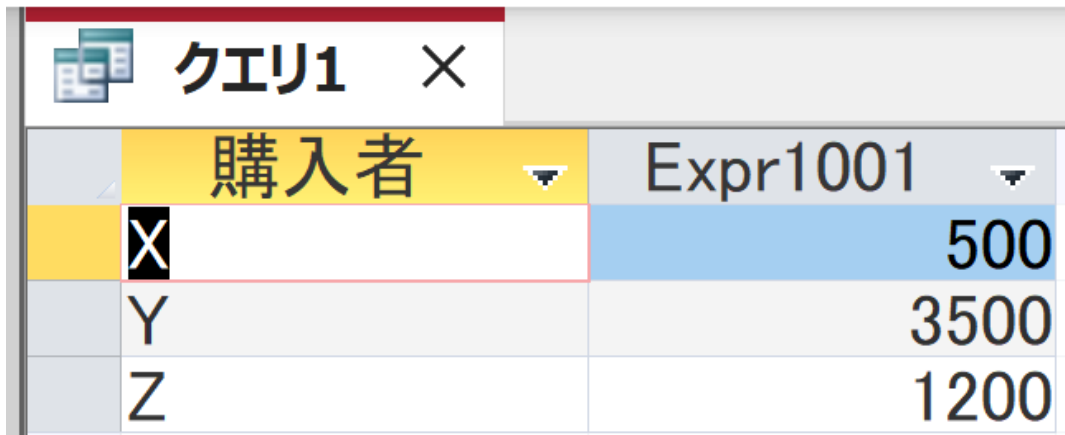
購入日時	購入者	Expr1002
2023/12/10 10:00:00		500
2023/12/20 12:00:00	Y	500
2024/01/05 9:00:00	Y	3000
2024/01/05 11:00:00	Z	1200

*

⑥ **SQLビュー**に、次のSQLを入れ、「**実行**」ボタンで、**SQL文**を実行。結果を確認

2つのテーブルを使い、購入者ごとに申し込みの合計金額を求める

```
SELECT 購入.購入者, SUM(購入.数量 * 商品.単価)
FROM 商品
INNER JOIN 購入 ON 商品.ID = 購入.商品ID
GROUP BY 購入.購入者;
```

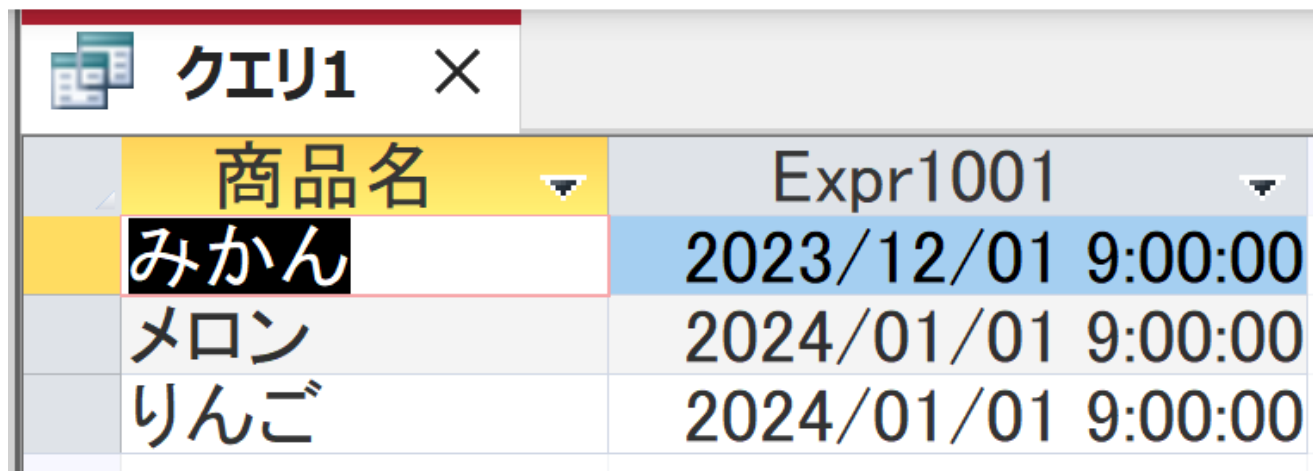


購入者	Expr1001
X	500
Y	3500
Z	1200

⑦ **SQL ビュー**に、次の SQL を入れ、「**実行**」ボタンで、**SQL文**を実行。結果を確認

各商品の最新の改訂日時を得ている

```
SELECT 商品名, MAX(改訂日時)
FROM 商品
GROUP BY 商品名;
```



商品名	Expr1001
みかん	2023/12/01 9:00:00
メロン	2024/01/01 9:00:00
りんご	2024/01/01 9:00:00

表示の幅はマウス操作で
変えることができる

⑧ SQL ビューに、次の SQL を入れ、「実行」ボタンで、SQL文を実行。結果を確認

購入テーブルを用いて、購入者が「Y」のすべての購入情報を得る

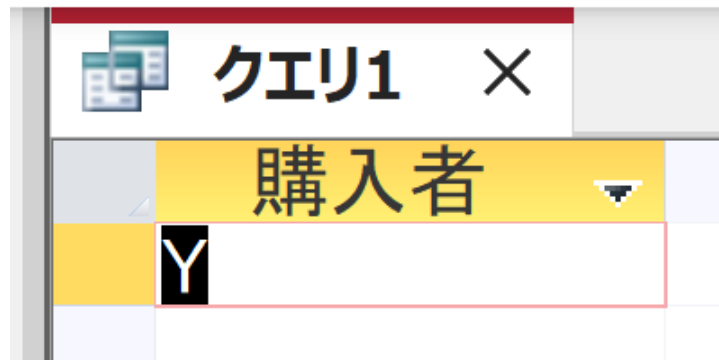
```
SELECT *  
FROM 購入  
WHERE 購入者 = 'Y';
```

ID	購入者	商品ID	数量	購入日時
2	Y	2	5	2023/12/20 12:00:00
3	Y	4	20	2024/01/05 9:00:00
*				

⑨ **SQL ビュー**に、次の SQL を入れ、「**実行**」ボタンで、**SQL文**を実行。結果を確認

商品名が「りんご」である商品を購入したすべての購入者
を得る

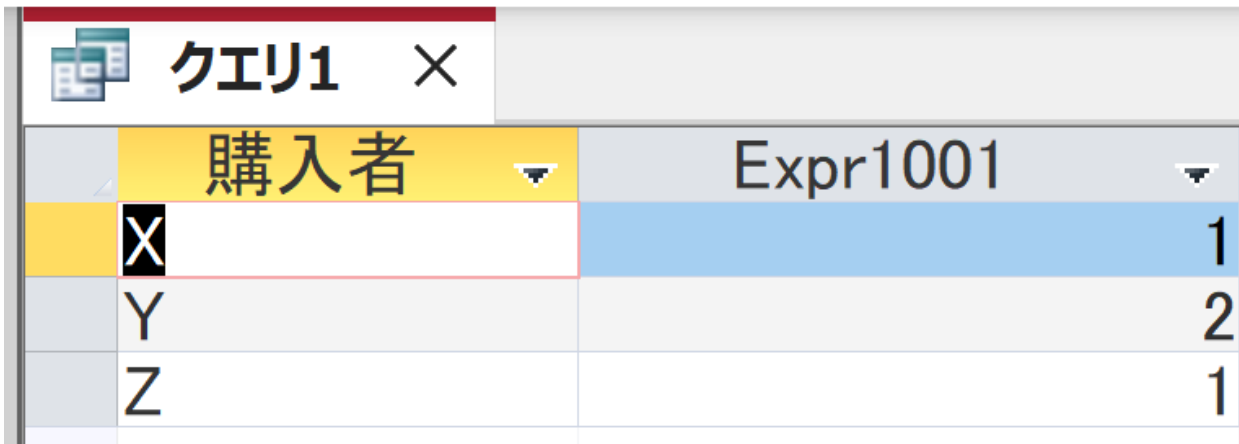
```
SELECT DISTINCT (購入.購入者)
FROM 購入
INNER JOIN 商品 ON 購入.商品ID = 商品.ID
WHERE 商品.商品名 = 'りんご';
```



⑩ SQL ビューに、次の SQL を入れ、「実行」ボタンで、SQL文を実行。結果を確認

購入テーブルを使用して、購入者ごとに、購入の回数を得る

```
SELECT 購入者, COUNT(*)  
FROM 購入  
GROUP BY 購入者;
```



The screenshot shows a query result window titled 'クエリ1 ×'. The window displays a table with two columns: '購入者' (Buyer) and 'Expr1001' (Count). The data is as follows:

購入者	Expr1001
X	1
Y	2
Z	1