## 8-1-1 文字の扱いについて

- ・キーワードの大文字と小文字は区別しない
- ・文字や日付情報はシングルクォートで囲む

```
Student/postgres@student

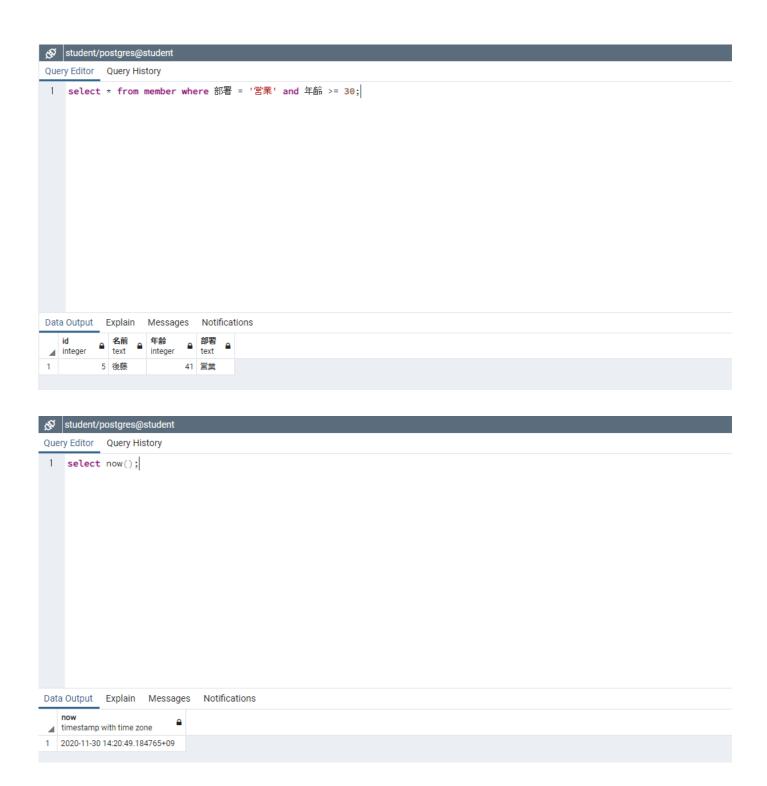
Query Editor Query History

1 create table member(
2 id integer,
4 名前 text,
5 年齡 integer,
6 部署 text
7 );
```

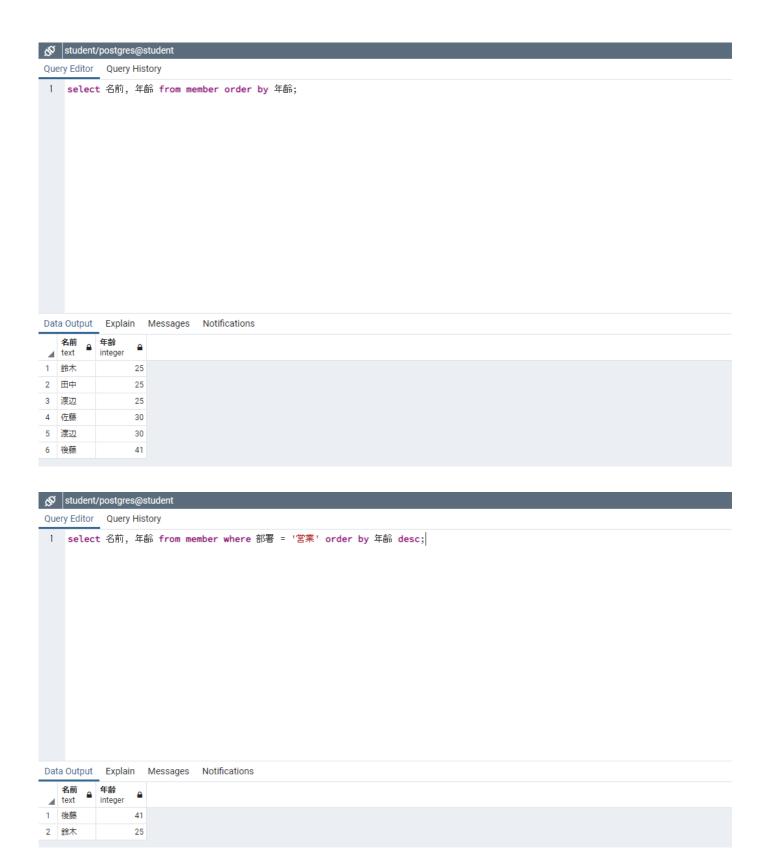
#### ※カラム名に、''は不要

```
student/postgres@student
Query Editor Query History
 1 insert into member
 3 values(1, '鈴木', 25, '営業'),(2, '佐藤', 39, '開発'),(2, '田中', 25, '開発'),(4, '渡辺', 39, '総務'), (5, '後藤', 41, '営業'), (6, '渡辺', 25, '開発');
 6 select * from member;
Data Output Explain Messages Notifications
 id 名前 年齡 部署 text
         1 鈴木
                       25 営業
2
         2 佐藤
                       30 開発
         2 田中
                       25 開発
         4 渡辺
4
                       30 総務
5
         5 後藤
                       41 営業
         6 渡辺
                       25 開発
```

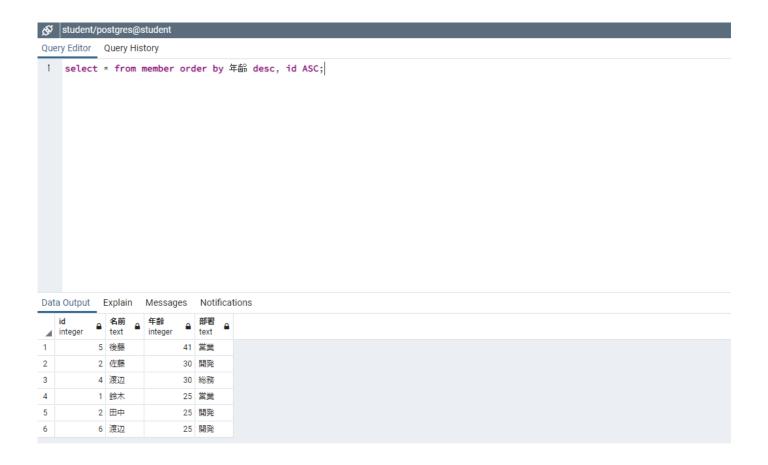
# 8-1-2 列(カラム)、テーブル、検索条件の指定



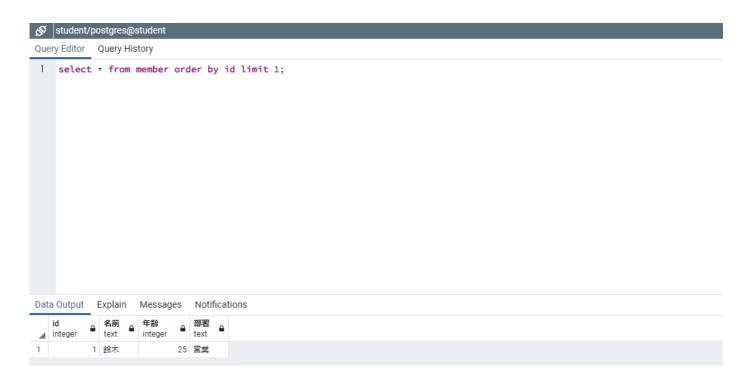
### **8-1-3 ORDER BY**



order byは左から順に処理される。1つ目の並べ替えで同じ順のものがあれば、2つ目の並べ替えが適用される。



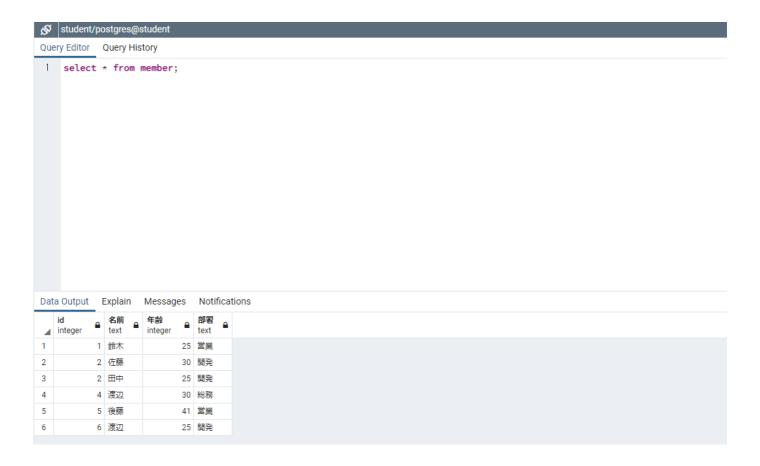
## 8-1-4 LIMIT ∠ OFFSET

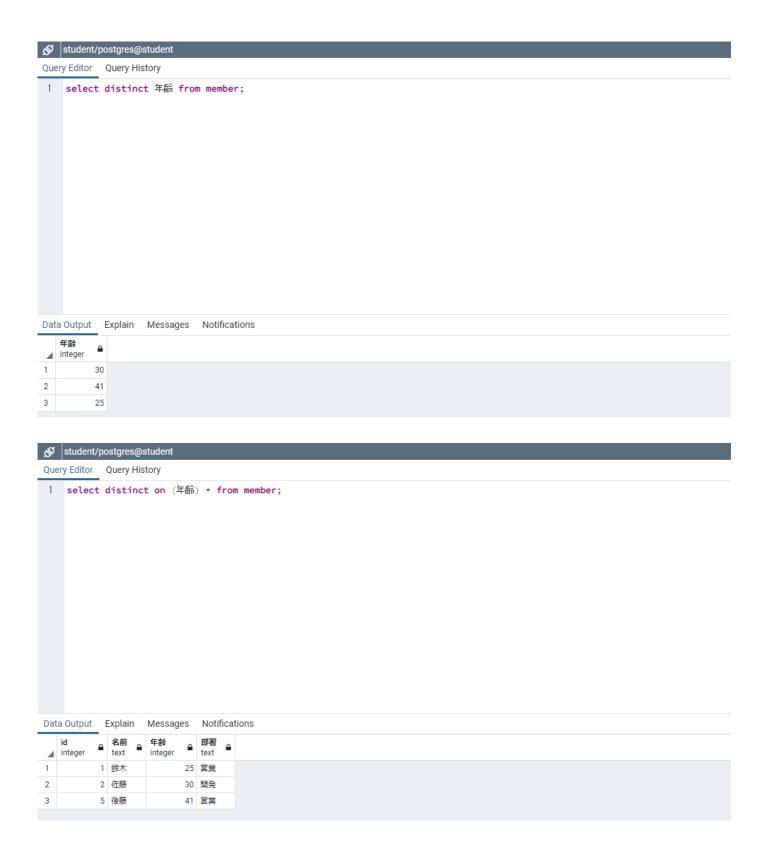




抽出されたデータの1件目をスキップし、2件目以降から3個のデータを取り出している。

### 8-1-5 DISTINCT





on (列名) を省略すれば、重複除去および抽出対象が年齢となり on (年齢) \* と指定すれば、年齢に対してのみ重複除去して、全列を抽出する。

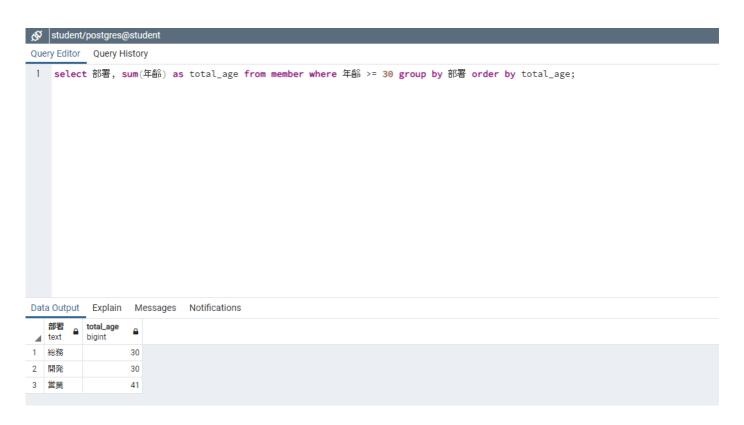
つまり、on (列名) を省略すると、重複除去・抽出の両方の対象となる。

また、DISTINCT ONでの実行結果はソートされる。 ORDER BYを併用した場合は、ORDER BYが優先となる。

## 8-1-6 GROUP BY & Having



集約関数を使用すると、列名を複数指定できる。



処理の順番: Where → GROUP BY → HAVING



部署でグループ化してから、最高年齢が30歳以下の部署に絞り、部署とsum(年齢)を取り出している。

### 8-1-7 副問い合わせ

SELECT (問い合わせ) → サブセレクトと呼ばれる。

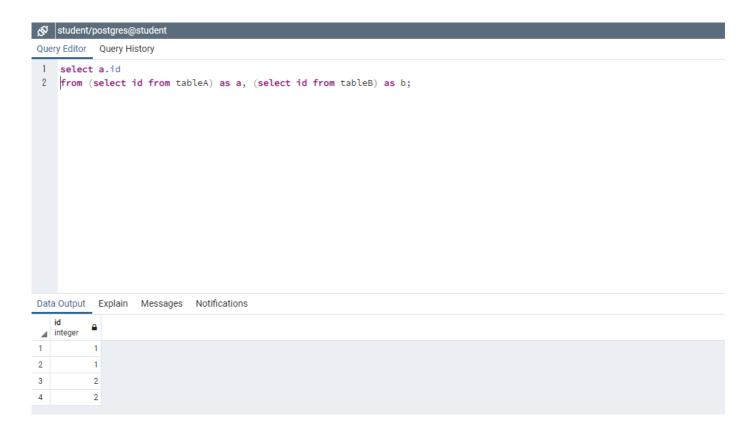




※副問い合わせの結果が2件以上になると、エラーとなるので注意すること。

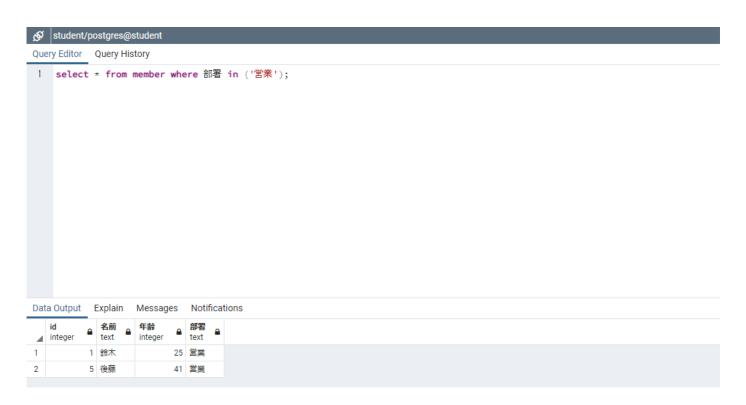


問い合わせ対象のテーブルが2つあるため、抽出結果も2つ分となる。



副問い合わせが2個以上ある場合には、エイリアス(asでつける別名)が必要となる。

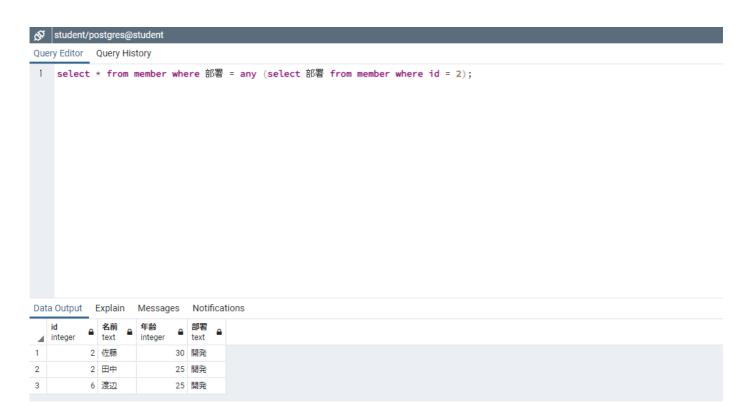
## 8-1-7 IN と NOT IN





### 8-1-9 ANY

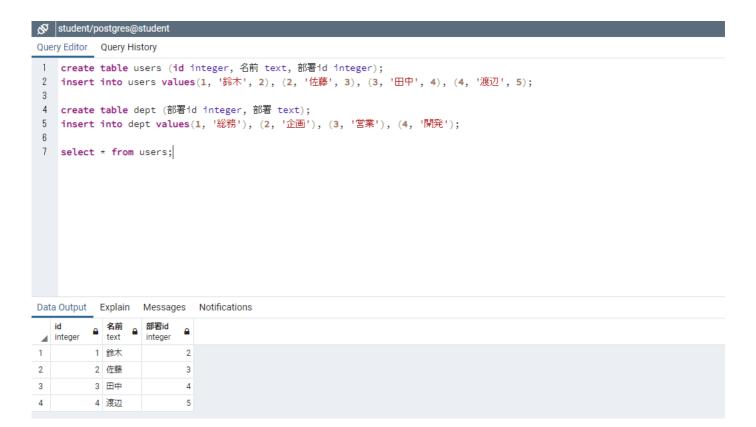
IN と同じ働きだが、副問い合わせの結果が必ず1列となる。

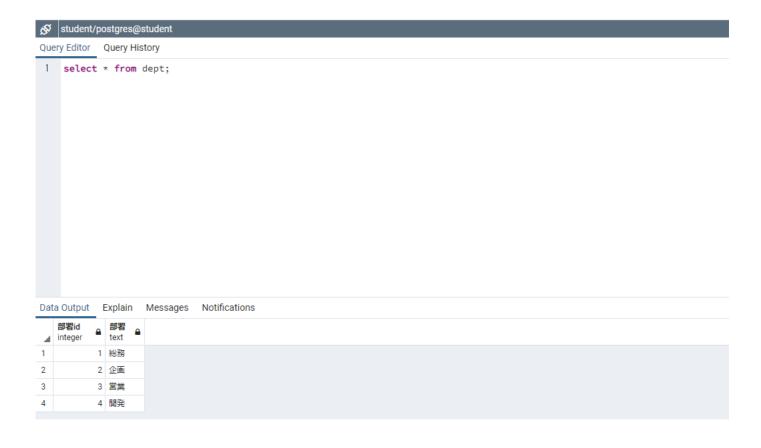


### 8-1-10 **BETWEEN**



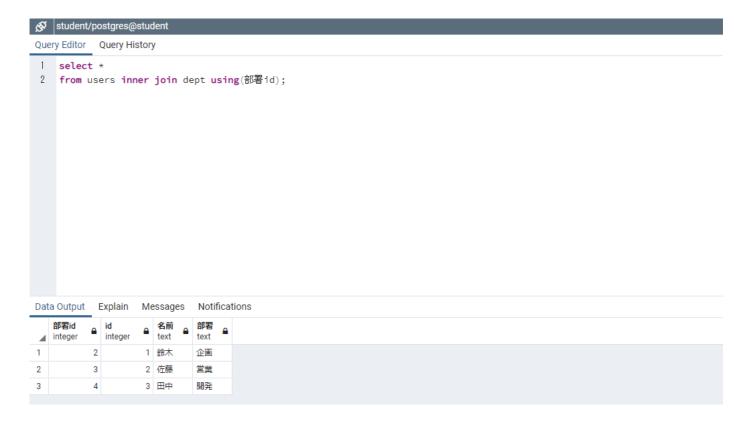
## 8-1-11 結合



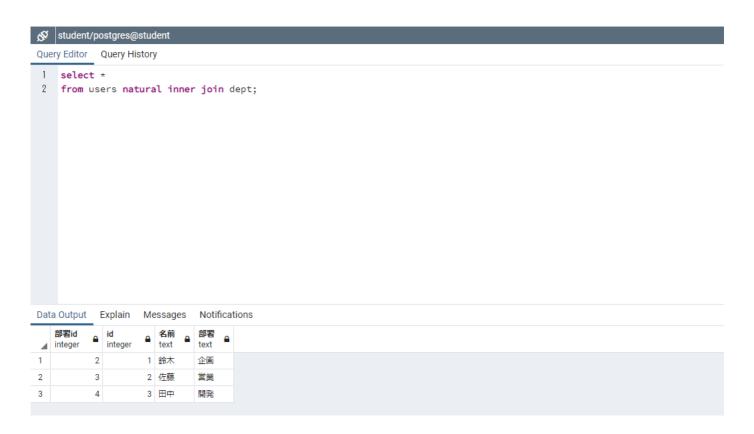


#### **INNER JOIN**

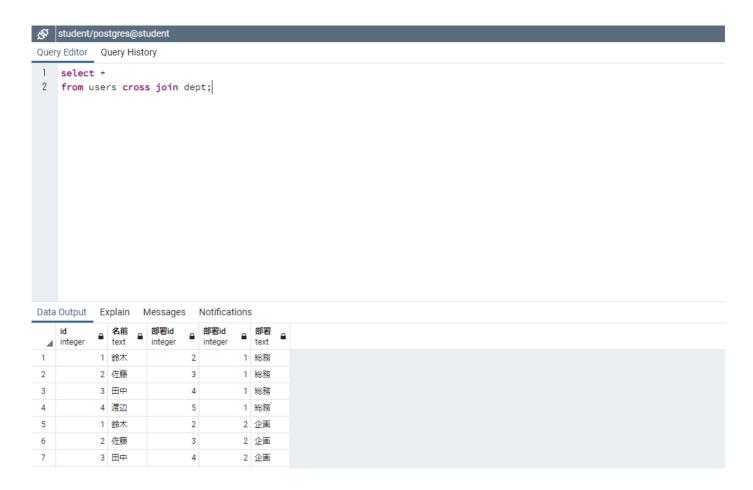




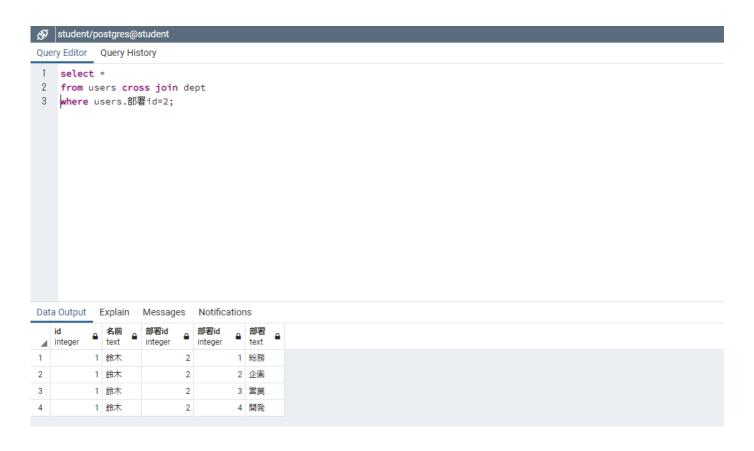
NATURALを用いた場合、結合に使用された列は一番左の列としてまとめられる。

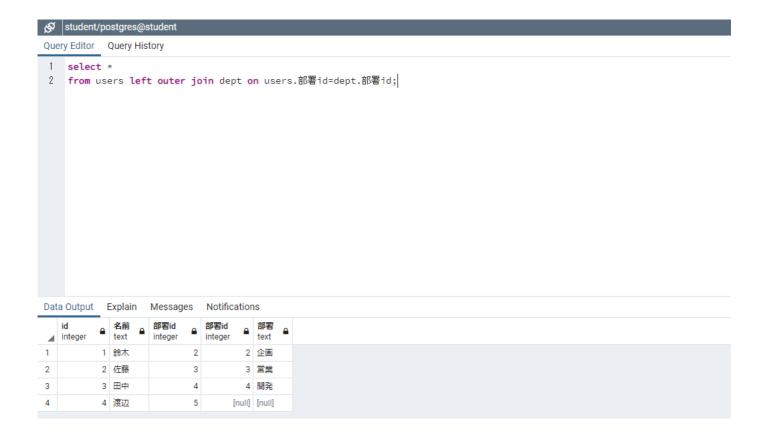


**CROSS JOIN** 

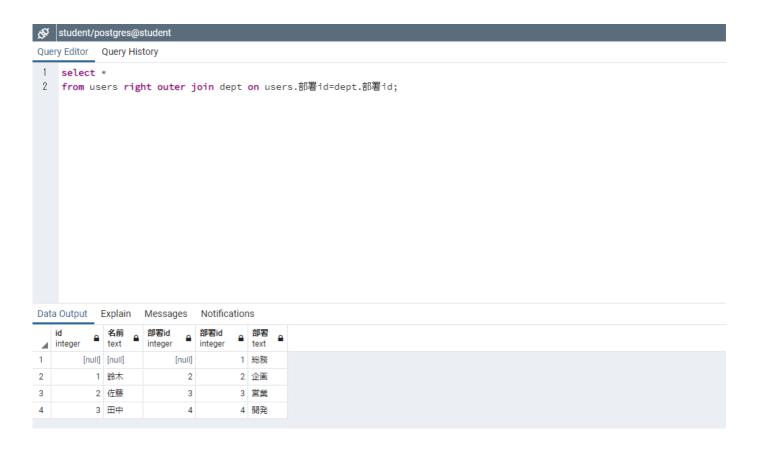


右から2番目の部署idを見ると、全ての組み合わせが結合されていることが分かりやすい。

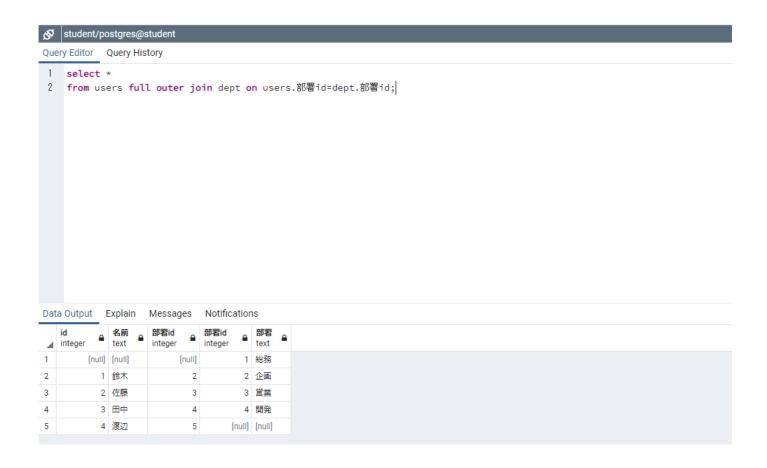




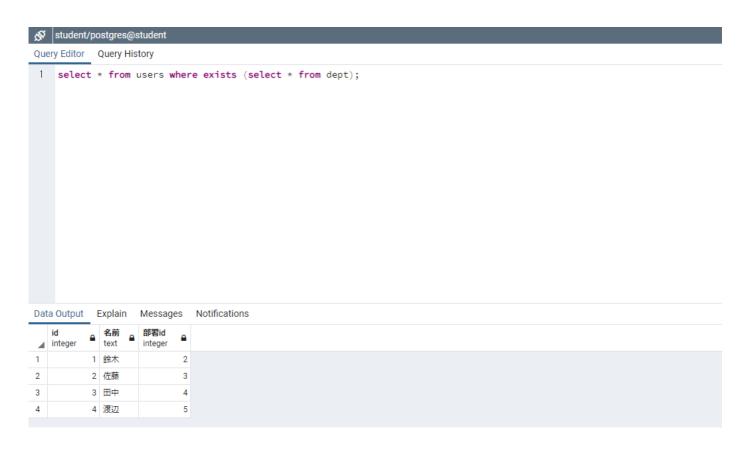
#### **RIGHT OUTER JOIN**

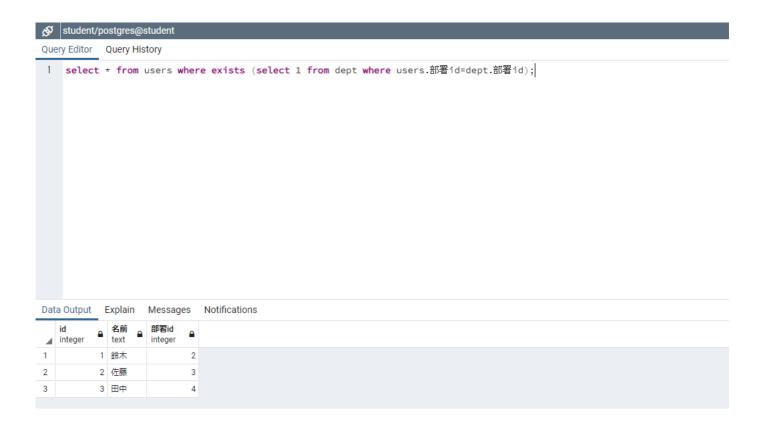


#### **FULL OUTER JOIN**



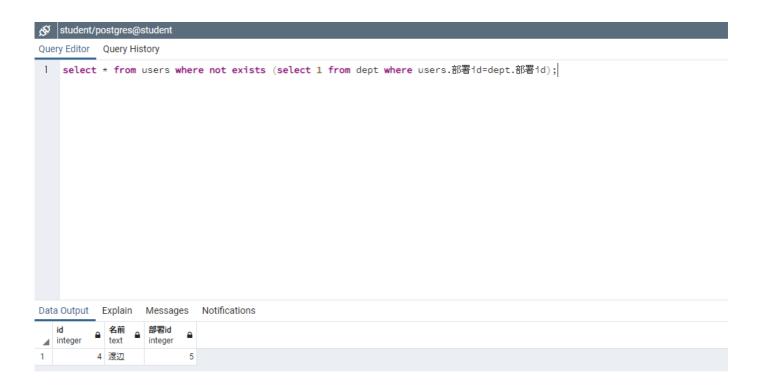
## 8-1-12 EXISTS ≥ NOT EXISTS



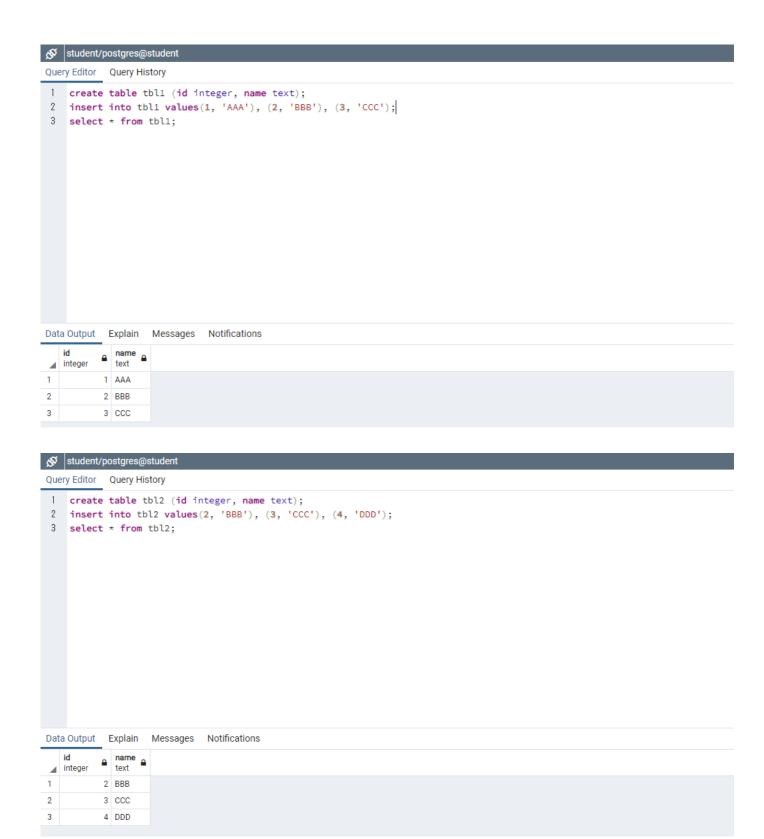


EXISTSは行を返すかどうかの指定しかしないため、副問い合わせのselectでは何を指定しても問題ない。 副問い合わせ内の条件に合致するものがあれば、usersの中から抽出するだけである。

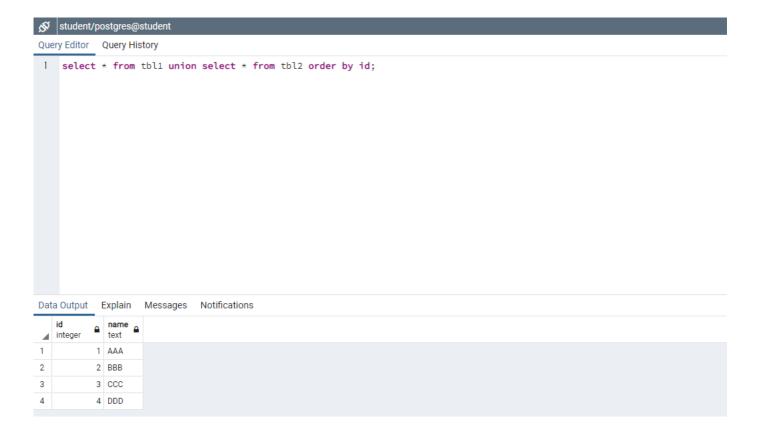
逆にNOT EXISTSは、条件に合致しないものを、usersの中から抽出する。



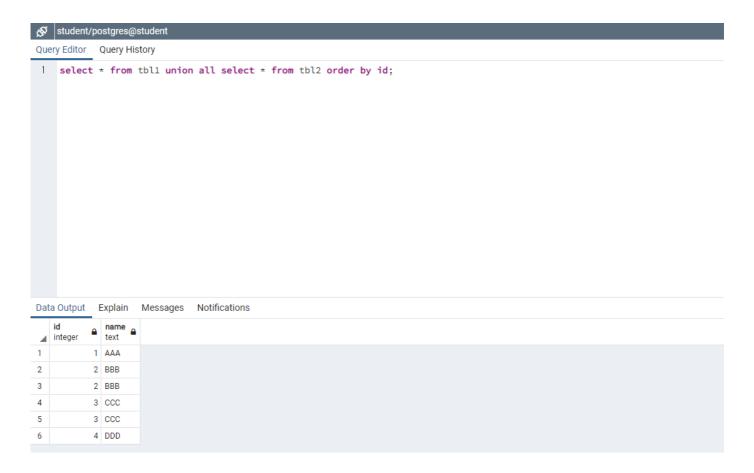
## 8-1-13 UNION/EXCEPT/INTERSECT



UNIONした結果に対して、ORDER BYを実行している。



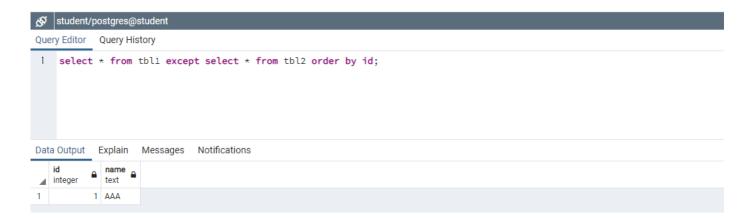
### ALLを付与すると、重複行がそのまま出力される。



EXCEPTは、2つのテーブルのうち、左側のテーブルにあるデータだけを抽出する。



INTERSECTは、2つのテーブルのうち、両方とも存在するデータのみを抽出する。



UNION/EXCEPT/INTERSECTを同時に使用した場合、最優先となるのはINTERSECTとなり それ以外は左から順に処理される。