

## OSS-DB Silver 技術解説セミナー

2023/02/05 開催

主題 一般知識

副題 OSS-DBの一般的特徴 【重要度：4】  
リレーショナルデータベースに関する一般知識 【重要度：4】

本日の講師

河原木 忠司（かわらぎただし）

## ■河原木忠司（かわらぎただし）

- 20数年ほど、講師/エンジニアとして活動しております。
- 最近では講師、研修コンサルティング、執筆業に従事させていただいております。
  - サーバーインフラ系のコース、セキュリティ系のコースを中心に担当させていただいております。
  - LinuC Lv.1/Lv.2のver.10試験開発にも協力させていただきました。  
<https://linuc.org/linuc/thanks.html>

## ■最近執筆したものの



「最短突破 LinuCレベル1  
合格教本 ver.10対応」  
(技術評論社)  
好評発売中です。

[https://gihyo.jp/book/  
2020/978-4-297-11527-2](https://gihyo.jp/book/2020/978-4-297-11527-2)



WebやDB系などの執筆を担当  
させていただきました

「標準テキスト CentOS8 構  
築・運用・管理パーフェクトガ  
イド[CentOS Stream対応]」  
(共著、SBクリエイティブ)  
[https://www.sbcr.jp/  
product/4815602567/](https://www.sbcr.jp/product/4815602567/)

## ■ OSS-DBとは



オープンソースのデータベースソフトウェア「PostgreSQL」を扱うことができる技術力の認定です。様々な分野でPostgreSQLの利用拡大が進む中でOSS-DBの認定を持つことは、自分のキャリアのアピールにもつながります。

- ✓ OSS-DB Goldは設計やコンサルティングができる技術力の証明

PostgreSQLについての深い知識を持ち、データベースの設計や開発のほか、パフォーマンスチューニングやトラブルシューティングまで行えることが証明できます
- ✓ OSS-DB Silverは導入や運用ができる技術力の証明

PostgreSQLについての基本的な知識を持ち、データベースの運用管理が行えるエンジニアとしての証明ができます
- ✓ 対象のバージョンはPostgreSQL 11

主題	副題
一般知識 (16%)	<ul style="list-style-type: none"> <li>• OSS-DBの一般的特徴【重要度:4】</li> <li>• リレーショナルデータベースに関する一般知識【重要度:4】</li> </ul>
運用管理 (52%)	<ul style="list-style-type: none"> <li>• インストール方法【重要度:2】</li> <li>• 標準付属ツールの使い方【重要度:5】</li> <li>• 設定ファイル【重要度:5】</li> <li>• バックアップ方法【重要度:7】</li> <li>• 基本的な運用管理作業【重要度:7】</li> </ul>
開発/SQL (32%)	<ul style="list-style-type: none"> <li>• SQL コマンド【重要度: 13】</li> <li>• 組み込み関数【重要度:2】</li> <li>• トランザクションの概念【重要度:1】</li> </ul>

今回のセミナーのテーマ  
8/50⇒16%

[https://oss-db.jp/outline/silver#questionnaire\\_range\\_silver](https://oss-db.jp/outline/silver#questionnaire_range_silver)

# OSS-DBの一般的な特徴

## ■ オープンソースソフトウェアで提供されているデータベースシステム



## ■ 主な製品

- PostgreSQL
- MySQL
- mariaDB など

OSS-DB(ver.2.0)試験では  
PostgreSQL 11を前提に出題

### DB-Engines Ranking

The DB-Engines Ranking ranks database management systems according to their popularity. The ranking is updated monthly.

Read more about the [method](#) of calculating the scores.



402 systems in ranking, January 2023

Rank			DBMS	Database Model	Score		
Jan 2023	Dec 2022	Jan 2022			Jan 2023	Dec 2022	Jan 2022
1.	1.	1.	Oracle +	Relational, Multi-model ⓘ	1245.17	-5.14	-21.72
2.	2.	2.	MySQL +	Relational, Multi-model ⓘ	1211.96	+12.56	+5.91
3.	3.	3.	Microsoft SQL Server +	Relational, Multi-model ⓘ	919.39	-4.96	-25.43
4.	4.	4.	PostgreSQL +	Relational, Multi-model ⓘ	614.85	-3.13	+8.29
5.	5.	5.	MongoDB +	Document, Multi-model ⓘ	455.18	-14.15	-33.38
6.	6.	6.	Redis +	Key-value, Multi-model ⓘ	177.56	-5.01	-0.43
7.	7.	7.	IBM Db2	Relational, Multi-model ⓘ	143.57	-3.05	-20.63

<https://db-engines.com/en/ranking>  
より



## ■OSSで提供

- BSDライセンスをベースとしたThe PostgreSQLライセンスにより提供
- OSSコミュニティによる提供
  - コミュニティによる豊富な情報提供
    - 公式サイト <https://www.postgresql.org/>
    - 日本PostgreSQLユーザ会 <https://www.postgresql.jp/>
      - » PostgreSQL 11文書(html) <https://www.postgresql.jp/document/11/html/>

## ■マルチプラットフォーム

- 様々なOS環境で動作 <https://www.postgresql.jp/download>

### ▼参考: 他DBMSとの比較

SQLの観点から「Oracle Database」「PostgreSQL」「MySQL」の特徴を整理しよう! | アシスト

[https://www.ashisuto.co.jp/tech-note/article/20141006\\_db.html](https://www.ashisuto.co.jp/tech-note/article/20141006_db.html)

## ■OSSライセンスの一種

- 非コピーレフト系

- 「コピーレフト」：再配布の際にも同じライセンスを適用
- GPLはコピーレフト系

- Apache Software LicenseもBSD系

- 再配布時、全てのソース、もしくは重要なファイルに著作権表示ならびに免責条項を表示



#OSS-DB

## ■最新版：2023/1現在

- 15.1
- 14.6
- 13.9
- 12.13
- **11.18**
- 10.23

メジャーバージョンが一緒だと、ファイルに互換性あり

「11」⇒メジャーバージョン  
「.18」⇒マイナーバージョン  
※ver.9までは「9.x」までがメジャーバージョン

### Releases

<https://www.postgresql.org/support/versioning/>

Version	Current minor	Supported	First Release	Final Release
15	15.1	Yes	October 13, 2022	November 11, 2027
14	14.6	Yes	September 30, 2021	November 12, 2026
13	13.9	Yes	September 24, 2020	November 13, 2025
12	12.13	Yes	October 3, 2019	November 14, 2024
11	11.18	Yes	October 18, 2018	November 9, 2023
10	10.23	No	October 5, 2017	November 10, 2022
9.6	9.6.24	No	September 29, 2016	November 11, 2021

ver.11の最終バージョンは、2023年11月の予定。  
※5年間サポート

マイナーアップデートは通常3ヶ月ごと

## ■ 様々なOSで動作する実行ファイルやソースファイルをダウンロード可能



- <https://www.postgresql.jp/download>
- <https://www.enterprisedb.com/downloads/postgres-postgresql-downloads>

PostgreSQL Version	Linux x86-64	Linux x86-32	Mac OS X	Windows x86-64	Windows x86-32
15.1	<a href="https://www.postgresql.org">postgresql.org</a>	<a href="https://www.postgresql.org">postgresql.org</a>			Not supported
14.6	<a href="https://www.postgresql.org">postgresql.org</a>	<a href="https://www.postgresql.org">postgresql.org</a>			Not supported
13.9	<a href="https://www.postgresql.org">postgresql.org</a>	<a href="https://www.postgresql.org">postgresql.org</a>			Not supported
12.13	<a href="https://www.postgresql.org">postgresql.org</a>	<a href="https://www.postgresql.org">postgresql.org</a>			Not supported
11.18	<a href="https://www.postgresql.org">postgresql.org</a>	<a href="https://www.postgresql.org">postgresql.org</a>			
9.6.24*					

このあとは、Linux (CentOS7) 環境にインストールしたPostgreSQL 11を利用して、デモ操作を行います。

■ PostgreSQLを商用システムに組み込んで販売したい。以下の説明からもっとも適切なものを1つ選びなさい。



#OSS-DB

- A) PostgreSQLを商用システムに組み込んで販売することは認められない。PostgreSQLをシステムに同梱せず、別途インストールする、といった対応をする必要がある。
- B) PostgreSQLの著作権を有するPostgreSQLグローバル開発チームと書面によるライセンス契約を結べば、その契約に従って販売することができる。
- C) システムの売上高に応じた一定の金額をPostgreSQLグローバル開発チームに納入することを条件に、販売することができる。
- D) 著作権表示を適切に行えば、販売することができる。
- E) PostgreSQLの利用や販売に関しては何も制限はないので、特に注意することではなく、自由に販売することができる。

■ PostgreSQLのライセンスについて適切な説明を2つ選びなさい。



#OSS-DB

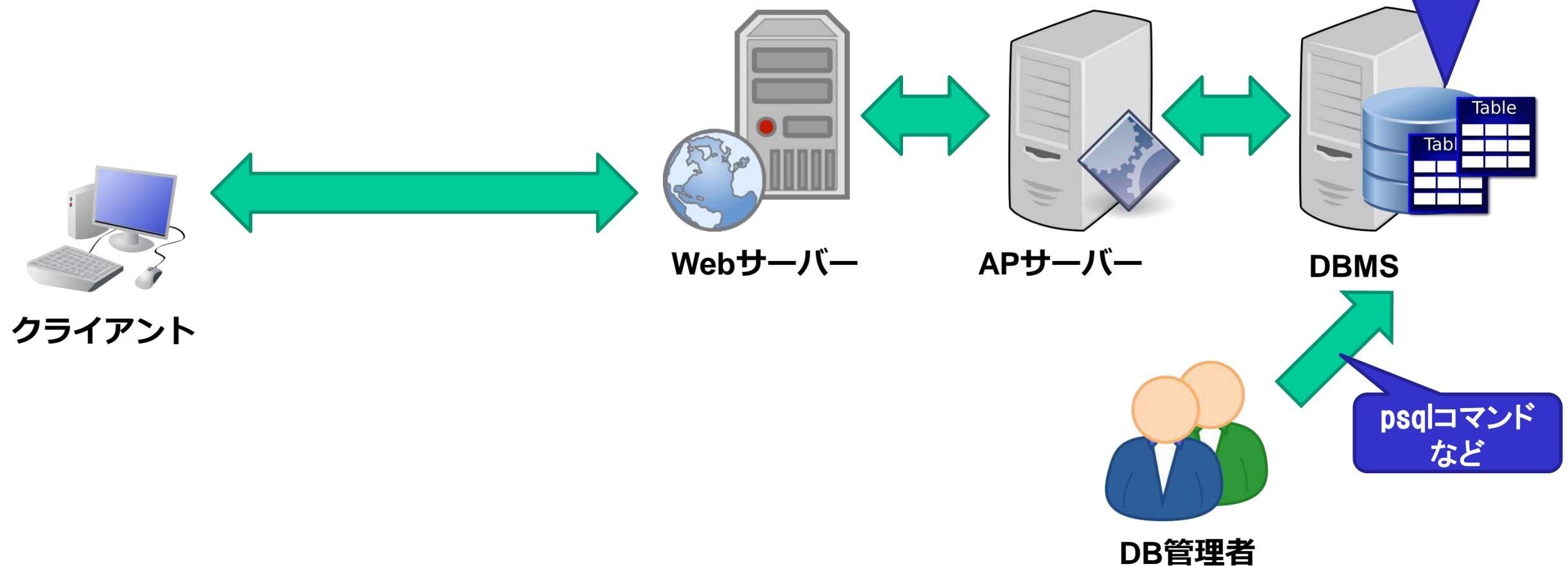
- A) Linuxカーネルと同じ、GPLに基づくライセンスを採用している。
- B) PostgreSQLライセンスという名前の、BSDに基づくライセンスを採用している。
- C) PostgreSQLのソースコードを改変して作成したプログラムを有償で販売することは認められていない。
- D) PostgreSQLのソースコードを改変して作成したプログラムを有償で販売する場合、改変部分を公開する義務がある。
- E) PostgreSQLのソースコードを改変して作成したプログラムを有償で販売することが認められており、このとき改変部分について公開する必要はない。

# リレーショナルデータベースに関する 一般知識

## ■ データベースを管理するシステム

- データを一元管理

## ■ 構成例：Webシステム

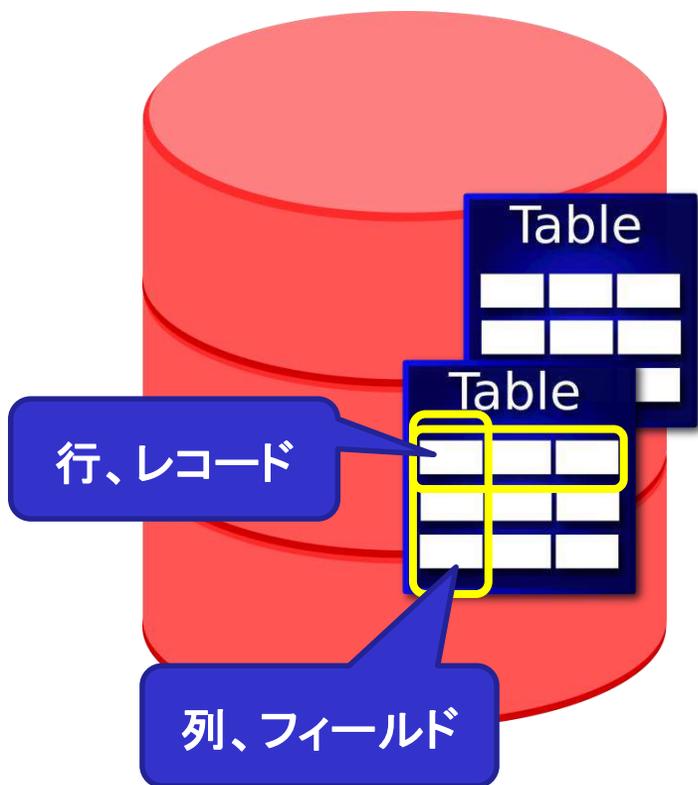


- データを操作するための環境を提供
  - SQLでデータ操作
  - データの独立性
  - アプリケーションからの接続
- 同時処理への対応
- 障害発生時の対応
  - トランザクション処理
  - バックアップ
- セキュリティ
  - DBMS独自の認証/アクセス制御



#OSS-DB

- リレーショナルデータベース (RDB) を管理するシステム
- データベース内に複数のテーブル (表) を構成
  - テーブルにデータを格納
- 複数のテーブルを関連付けて、データを管理



▼データベースの作成

```
$ createdb testdb
```

▼データベースに接続

```
$ psql -d testdb
```

```
testdb=# \d
```

リレーション一覧			
スキーマ	名前	型	所有者
public	test	テーブル	testuser

▼テーブルの作成

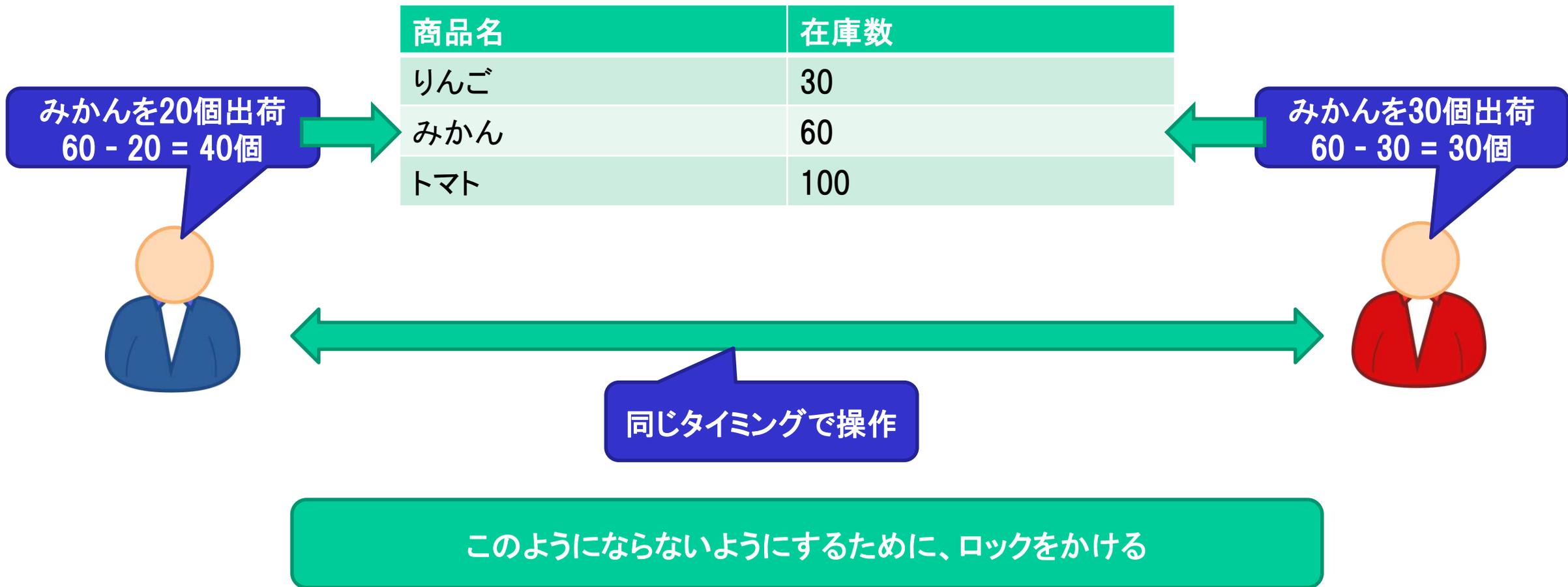
```
create table test (
id integer primary key,
name varchar(20)
);
```

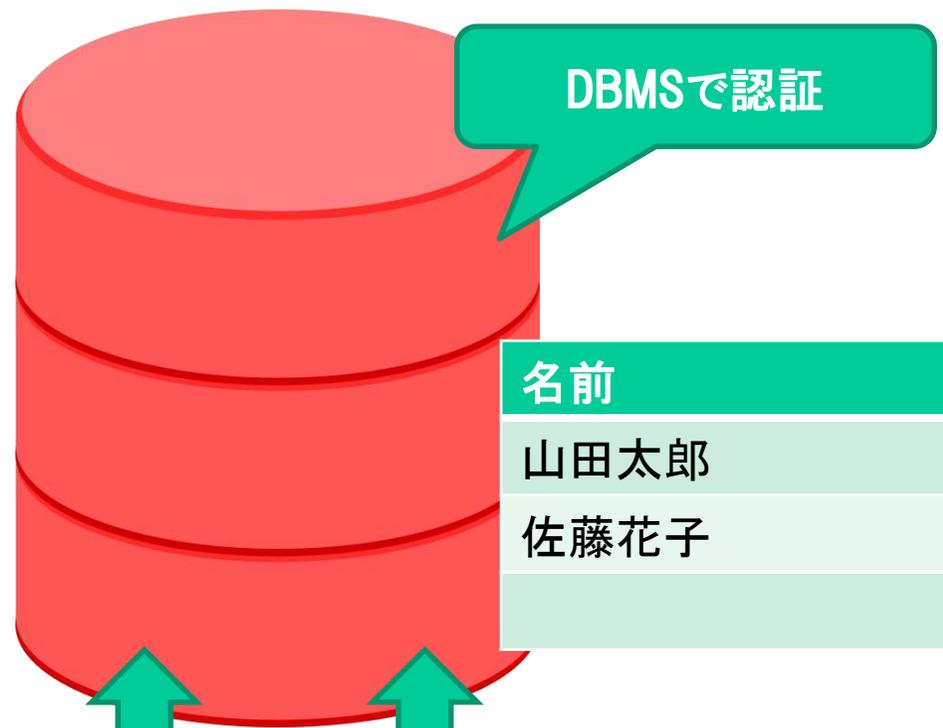
テーブル作成時に  
列名、データ型、制  
約も指定

- RDBMSにおいて、データの操作や定義を行うための言語
- ANSI、ISOにより標準化
  - 様々なRDBMSで利用可能
    - RDBMSごとに特有の文法などもある
- 主な規格
  - SQL86
  - SQL89
  - :
  - SQL:2008
  - SQL:2011
  - SQL:2016

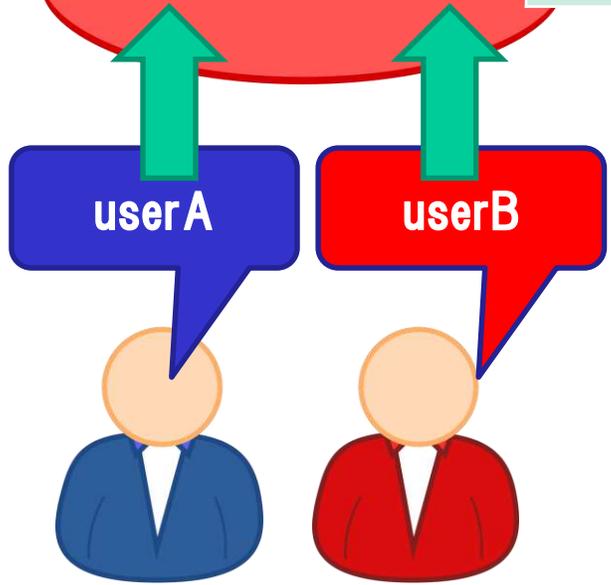
▼参考:SQLへの準拠  
 PostgreSQL 11.5文書: 付録D SQLへの準拠  
<https://www.postgresql.jp/document/11/html/features.html>

分類	説明	主なステートメント
DML文	<u>データを操作</u> テーブル内のレコードを参照、挿入、更新、削除	<ul style="list-style-type: none"> <li>• SELECT</li> <li>• INSERT</li> <li>• UPDATE</li> <li>• DELETE</li> </ul>
DDL文	<u>データを定義</u> テーブルを作成、変更、破棄	<ul style="list-style-type: none"> <li>• CREATE</li> <li>• ALTER</li> <li>• DROP</li> </ul>
DCL文	<u>データを制御</u> リソースへのアクセスを制御、トランザクション処理を確定、ロールバック	<ul style="list-style-type: none"> <li>• GRANT</li> <li>• REVOKE</li> <li>• COMMIT</li> <li>• ROLLBACK</li> </ul>

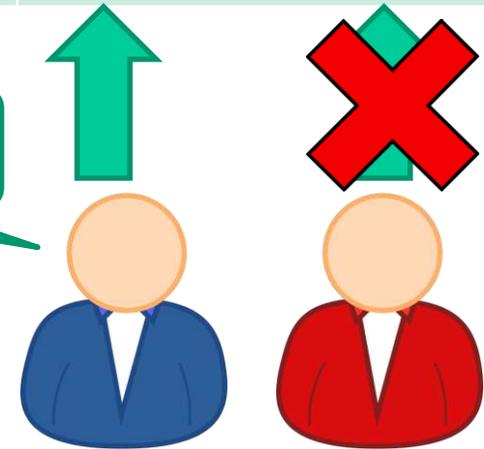




名前	電話番号	email	カード番号
山田太郎	0451112222	yamada@example.com	35...
佐藤花子	0113334444	info@sato.local	46...



テーブルや列などの  
単位で権限制御





■SQLに関する説明として正しいものを3つ選びなさい。

- A) DCLはData Creation Language（データ作成言語）の略で、CREATE TABLEやINSERTがそれに該当する。
- B) DCLはData Control Language（データ制御言語）の略で、GRANTやCOMMITがそれに該当する。
- C) DDLはData Definition Language（データ定義言語）の略で、CREATE TABLEやDROP TABLEがそれに該当する。
- D) DDLはData Deleting Language（データ削除言語）の略で、DELETEやDROP TABLEがそれに該当する。
- E) DMLはData Manipulation Language（データ操作言語）の略で、SELECTやINSERTがそれに該当する。
- F) DMLはData Modification Language（データ修正言語）の略で、ALTER TABLE や UPDATE がそれに該当する。



## ■データベースの設計

- 概念設計、論理設計、物理設計といった手順がある
- あるデータをデータベースシステム上で扱えるように設計
  - 前提：データベースシステム上のテーブルにデータを格納  
⇒テーブルを作成しなければならない

```
create table test (
id integer primary key,
name varchar(20)
);
```



どのような列を定義するか、主キー列はどこか。

id	name

数値データを挿入  
主キー列

文字データを挿入

データ型などを含む設計の詳細については、以下を参照。

【オンライン開催】『OSS-DB Exam Silver 技術解説無料セミナー』7/25(日)開催結果のご報告

<https://oss-db.jp/event/20210725>

## ■テーブルの中でレコードを特定できる列

- ほかのテーブルから参照するとき主キー列を指定

id	name	email	address

id列を用意し、特定

同姓同名のユーザーがいるかも...

同じ住所のユーザーがいるかも...

## ■列に対し、「primary key」制約を指定

- 次の制約を課す
  - 同じ値を入力できない
  - 空白を入力できない

## ■ 次のようなレコードが構成されているとき・・・

```
postgres=# select * from test;
```

id	name	email	address
1	Yamada Taro	yamada@example.com	Tokyo
2	Sato Hanako	info@sato.com	Yokohama
3	Yamada Taro		Yokohama

id列が主キー

(3行)

## ■ 次のようなレコードは挿入できない

```
postgres=# insert into test values(3,'Yoshida','yoshida@example.com','Tokyo');
```

**ERROR: 重複キーが一意性制約"test\_pkey"に違反しています**

**DETAIL: キー (id)=(3) はすでに存在します。**

```
postgres=# insert into test values(NULL,'Yoshida','yoshida@example.com','Tokyo')
```

```
;
```

**ERROR: 列"id"内のNULL値はNOT NULL制約違反です**

**DETAIL: 失敗した行は(null, Yoshida, yoshida@example.com, Tokyo)を含みます**

## ■正規化とは

- データの重複をなくし整合的にデータを取り扱えるようにデータベースを設計すること
- 正規化を行っておくと、データの追加・更新・削除などに伴うデータの不整合や喪失が起きるのを防ぎ、**メンテナンスの効率を高める**ことができる。

OSS-DB公式ページ：第4回 データベースの正規化  
[https://oss-db.jp/dojo/dojo\\_info\\_04](https://oss-db.jp/dojo/dojo_info_04) より

## ■複数のテーブルを関連付けて、整合性のとれたデータ管理を行う

商品名	売上額	担当者	店舗
りんご	800	斉藤	荒川店
みかん	500	田中	横浜店
みかん	2500	田中	横浜店
みかん	5000	斎藤	荒川店

誤入力

重複入力による  
無駄



商品名	売上額	担当者ID
りんご	800	101
みかん	500	102
みかん	2500	102
みかん	5000	101

担当者ID	担当者	店舗
101	斉藤	荒川店
102	田中	横浜店

担当者情報は  
別テーブルで管  
理し、効率化



日時	商品名	単価	個数	顧客名	顧客住所	顧客電話番号
2023/1/10 11:10	りんご	80	10	荒川商店	東京都荒川区1-1-1	03-1111-2222
	みかん	50	20			
2023/1/10 13:40	みかん	50	10	文京食堂	東京都文京区2-2-2	03-3333-4444
2023/1/11 9:00	トマト	100	5	レストラン横浜	神奈川県横浜市3-3	045-555-6666
	たまねぎ	70	10			
	りんご	80	10			
2023/1/11 9:00	みかん	50	20	荒川商店	東京都荒川区1-1-1	03-1111-2222

テーブルとして構成できない

日時	商品名	単価	個数	商品名	単価	個数	商品名	単価	個数	顧客名	顧客住所	顧客電話番号
2023/1/10 11:10	りんご	80	10	みかん	50	20				荒川商店	東京都荒川区1-1-1	03-1111-2222
2023/1/10 13:40	みかん	50	10							文京食堂	東京都文京区2-2-2	03-3333-4444
2023/1/11 9:00	トマト	100	5	たまねぎ	70	10	りんご	80	10	レストラン横浜	神奈川県横浜市3-3	045-555-6666
2023/1/11 9:00	みかん	50	20							荒川商店	東京都荒川区1-1-1	03-1111-2222

■ 繰り返し項目があった場合、個々のレコードとして構成できるようにする。

日時	商品名	単価	個数	顧客名	顧客住所	顧客電話番号
2023/1/10 11:10	りんご	80	10	荒川商店	東京都荒川区1-1-1	03-1111-2222
2023/1/10 11:10	みかん	50	20	荒川商店	東京都荒川区1-1-1	03-1111-2222
2023/1/10 13:40	みかん	50	10	文京食堂	東京都文京区2-2-2	03-3333-4444
2023/1/11 9:00	トマト	100	5	レストラン横浜	神奈川県横浜市3-3	045-555-6666
2023/1/11 9:00	たまねぎ	70	10	レストラン横浜	神奈川県横浜市3-3	045-555-6666
2023/1/11 9:00	りんご	80	10	レストラン横浜	神奈川県横浜市3-3	045-555-6666
2023/1/11 9:00	みかん	50	20	荒川商店	東京都荒川区1-1-1	03-1111-2222

独立したレコードとして構成

## ■重複している項目を別テーブルで構成し、効率的に管理

取引ID	日時	顧客名	顧客住所	顧客電話番号
1	2023/1/10 11:10	荒川商店	東京都荒川区1-1-1	03-1111-2222
2	2023/1/10 13:40	文京食堂	東京都文京区2-2-2	03-3333-4444
3	2023/1/11 9:00	レストラン横浜	神奈川県横浜市3-3	045-555-6666
4	2023/1/11 9:00	荒川商店	東京都荒川区1-1-1	03-1111-2222

取引ID	明細ID	商品名	単価	個数
1	1	りんご	80	10
1	2	みかん	50	20
2	1	みかん	50	10
3	1	トマト	100	5
3	2	たまねぎ	70	10
3	3	りんご	80	10
4	1	みかん	50	20

## ■重複している項目を別テーブルで構成し、効率的に管理

取引ID	日時	顧客ID
1	2023/1/10 11:10	1
2	2023/1/10 13:40	2
3	2023/1/11 9:00	3
4	2023/1/11 9:00	1

顧客ID	顧客名	顧客住所	顧客電話番号
1	荒川商店	東京都荒川区1-1-1	03-1111-2222
2	文京食堂	東京都文京区2-2-2	03-3333-4444
3	レストラン横浜	神奈川県横浜市3-3	045-555-6666

取引ID	明細ID	商品ID	個数
1	1	f01	10
1	2	f02	20
2	1	f02	10
3	1	v01	5
3	2	v02	10
3	3	f01	10
4	1	f02	20

商品ID	商品名	単価
f01	りんご	80
f02	みかん	50
v01	トマト	100
v02	たまねぎ	70



```
drop table if exists tbl_customer;
create table tbl_customer (
    customer_id integer primary key,
    customer_name varchar(20),
    address varchar(50),
    telno varchar(12)
);
insert into tbl_customer values(1,'荒川商店','東京都荒川区1-1-1','03-1111-2222');
insert into tbl_customer values(2,'文京食堂','東京都文京区2-2-2','03-3333-4444');
insert into tbl_customer values(3,'レストラン横浜','神奈川県横浜市3-3','045-555-6666');
```

```
drop table if exists tbl_product;
create table tbl_product (
    product_id varchar(3) primary key,
    product_name varchar(20),
    price numeric
);
insert into tbl_product values('f01','りんご',80);
insert into tbl_product values('f02','みかん',50);
insert into tbl_product values('v01','トマト',100);
insert into tbl_product values('v02','たまねぎ',70);
```

```
drop table if exists tbl_tran;
create table tbl_tran (
  tran_id integer primary key,
  tran_date timestamp,
  customer_id integer references tbl_customer(customer_id)
);
```

```
insert into tbl_tran values (1,'2023-1-10 11:10:00',1);
insert into tbl_tran values (2,'2023-1-10 13:40:00',2);
insert into tbl_tran values (3,'2023-1-11 9:00:00',3);
insert into tbl_tran values (4,'2023-1-11 9:00:00',1);
```



外部キー制  
約を設定

```
drop table if exists tbl_tran_detail;
create table tbl_tran_detail (
  tran_id integer,
  detail_id integer,
  product_id varchar(3) references tbl_product(product_id),
  quantity integer,
  primary key(tran_id,detail_id)
);
```

```
insert into tbl_tran_detail values(1,1,'f01',10);
insert into tbl_tran_detail values(1,2,'f02',20);
insert into tbl_tran_detail values(2,1,'f02',10);
insert into tbl_tran_detail values(3,1,'v01',5);
insert into tbl_tran_detail values(3,2,'v02',10);
insert into tbl_tran_detail values(3,3,'f01',10);
insert into tbl_tran_detail values(4,1,'f02',20);
```



#OSS-DB

- ほかのテーブルの情報を参照する際、整合性のとれたデータ管理ができるよう制約を課す



#OSS-DB

取引ID	日時	顧客ID	顧客ID	顧客名	顧客住所	顧客電話番号
1	2023/1/10 11:10	1	1	荒川商店	東京都荒川区1-1-1	03-1111-2222
2	2023/1/10 13:40	2	2	文京食堂	東京都文京区2-2-2	03-3333-4444
3	2023/1/11 9:00	3	3	レストラン横浜	神奈川県横浜市3-3	045-555-6666
4	2023/1/11 9:00	1				
5	2023/1/24 17:00	4				



主キー側テーブルに存在しないデータは入力できない

外部キー側テーブルで参照済みのレコードは削除できない

- データベースの正規化について最も適切な説明を1つ選びなさい。
  - A) データサイズを一定にすることでアクセス性能の向上を図る。
  - B) データ検索パスを標準化することでアクセス性能の向上を図る。
  - C) データの重複を排除することで、データ保守を容易にする。
  - D) データの依存関係を単純化することで、データ検索を容易にする。
  - E) テーブルの構造を標準化することで、データのバックアップを容易にする。



#OSS-DB

■ 以下は売上テーブルの一部だが、これを正規化したい。最も重要なものを2つ選びなさい。



日付	顧客名	顧客所在地	品目1	数量1	品目2	数量2	品目3	数量3
2月1日	淀橋	千代田区一番町	テレビ	5	PC	3	カメラ	5
2月1日	山田	港区麻布台	PC	5	ラジオ	2		
2月1日	小島	千代田区内幸町	冷蔵庫	1	洗濯機	4	電池	100
2月2日	野島	千代田区一番町	テレビ	10				
2月5日	山田	港区麻布台	マウス	10	DVD	20	CD	20
2月8日	小島	千代田区内幸町	カメラ	10				

- A) 日付に2/1が3回現れるなど重複があるので、別のテーブルに分離する。
- B) 同じ顧客が繰り返し現れるので、これを別のテーブルに分離する。
- C) 顧客名がわかれば顧客所在地がわかるので、この組み合わせを別のテーブルに分離する。
- D) 淀橋と野島は所在地が重複しているので、所在地を別のテーブルに分離する。
- E) 品目と数量の組み合わせを別のテーブルに分離する。

ご参加いただきありがとうございました。

 #OSS-DB

時間内に対応できなかったご質問については、後日動作公開の際に追加させていただきます。

## ■ サンプル問題の解答

- 問題の引用元⇒[https://oss-db.jp/sample/silver\\_knowledge\\_0](https://oss-db.jp/sample/silver_knowledge_0)

No.	解答
1	D
2	B, E
3	B, C, E
4	C
5	C, E