データサイエンス応用コース

構造化データ・蓄積・加工（1）

2021/03/04

大阪大学 特任准教授

shimokawa@sigmath.es.osaka-u.ac.jp 下川和郎

講義の概要

- DB アプリケーション開発の実際
  - なぜデータベース？ なぜ SQL ？
  - 開発手法の問題点
- SQL 入門
  - SQL コマンド基礎
    - SELECT、SORT、主キー
  - データベース正規化

講義の概要

- データベース設計
  - SQL コマンド応用
    - テーブル結合、副問い合わせ
    - データ入力インデックス作成
- 実行制御・セキュリティー
    - トランザクション、ロールバック、デッドロック
    - SQL インジェクション

なぜ SQL を学ぶのか？

- 文書作成ソフト（MS-Word等）を使ってもデータベース（DB）構築＆検索は可能
- テキストファイルとエディタ、手続き型言語（Perl, C）を使ってもDB構築は可能

だが…

- 大規模データのアクセスではデータベース管理システム（DBMS) が有効
- 多人数での利用（検索、書き換え、秘匿管理等）では DBMS が有効
- DBMS を他の用途に利用するためには SQL の知識が不可欠

PC で扱えるデータのサイズは？ Ex. MS Excel

Microsoft | サポート Microsoft 365 Office Windows Surface Xbox セール

Excel の仕様と制限

Excel for Microsoft 365, Excel 2019, Excel 2016, Excel 2013, Excel 2010, Excel 2007

ワークシートとブックの仕様と制限

機能	最大数
開くことのできるブック数	使用可能メモリとシステム リソースに依存
ワークシートの行数と列数	1,048,576 行、16,384 列
列の幅	255 文字

Web データベース 例
（中古車検索）

Web と組み合わせる利用方法が多い

https://www.goo-net.com/ より

メーカー別、車種別検索．写真掲載．　新車販売が終了した中古車も対象

Web データベース 例 （中古車検索）

県別集計

https://www.goo-net.com/ より

年式

走行距離

販売地域

車検残

https://www.goo-net.com/ より

車体色で探す？？？

- 白
  - パールホワイト
  - ピュアホワイトパール
  - フェザーホワイト
  - サテンホワイトパール
  - シャイニングホワイトパール
  - スマッシュホワイト
  - クリスタルパールマイカ
  - ゴールドパールクリスタルシャイン
  - アイボリーホワイト
  - ミネラルホワイト
  - ソニッククォーツ
  - アークティックホワイト
  - サミットホワイト
  - バーチホワイト
  - アルピンホワイト
  - …

白色の車体を検索するためには
これだけの情報を別テーブルで保有

Web データベース 例 （レコメンド機能）

二次元表に使われる直接的な関連性のある情報とは関係のない情報を検索・表示

http://www.amazon.com/ より

Web データベース 例 （トラベル）

8/1(土)

【身体に優しい北海道素材の朝食付】さき楽☆５日前までの予約でHappy！【空港までの送迎あり】

洋室・ツイン・バス・トイレ付

■禁煙■ツインルーム（ベッド幅122cm×2台）

■客室内にバス＆シャワートイレ設置　■ホテル仕様ポケットコイルベッド導入■■無料Wi-fi環境＆部屋の明かり調節OK

予約部屋数：1部屋

食事条件　昼食：0回　夕食：0回　翌朝食：1回

▼施設詳細

8/2(日)

【楽パックスペシャル】【バイキングプラン】十勝の美味しさをぎゅっと詰め込んだビュッフェ

和室・バス・トイレ付

和室８畳（田園側／禁煙）

当館の標準タイプのお部屋です。 お部屋の眺望は指定できません。

予約部屋数：1部屋

食事条件　昼食：0回　夕食：1回　翌朝食：1回

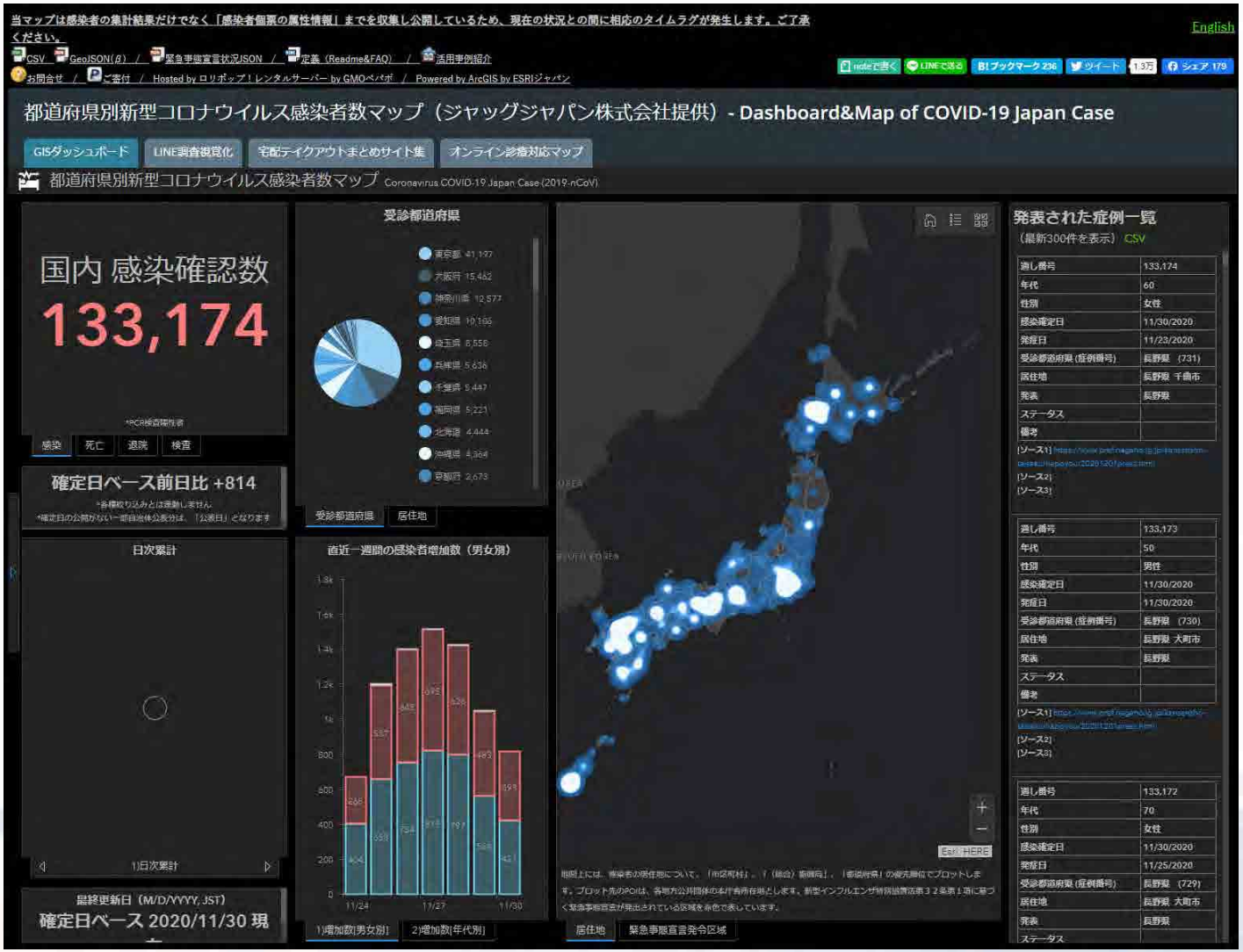
▼施設詳細

旅行計画：xx 日に xx 市内で空いている部屋のリストを表示．選択中に予約枠がなくなる可能性 → その後の処理

https://travel.rakuten.co.jp/ より

都道府県別新型コロナウィルス感染者数

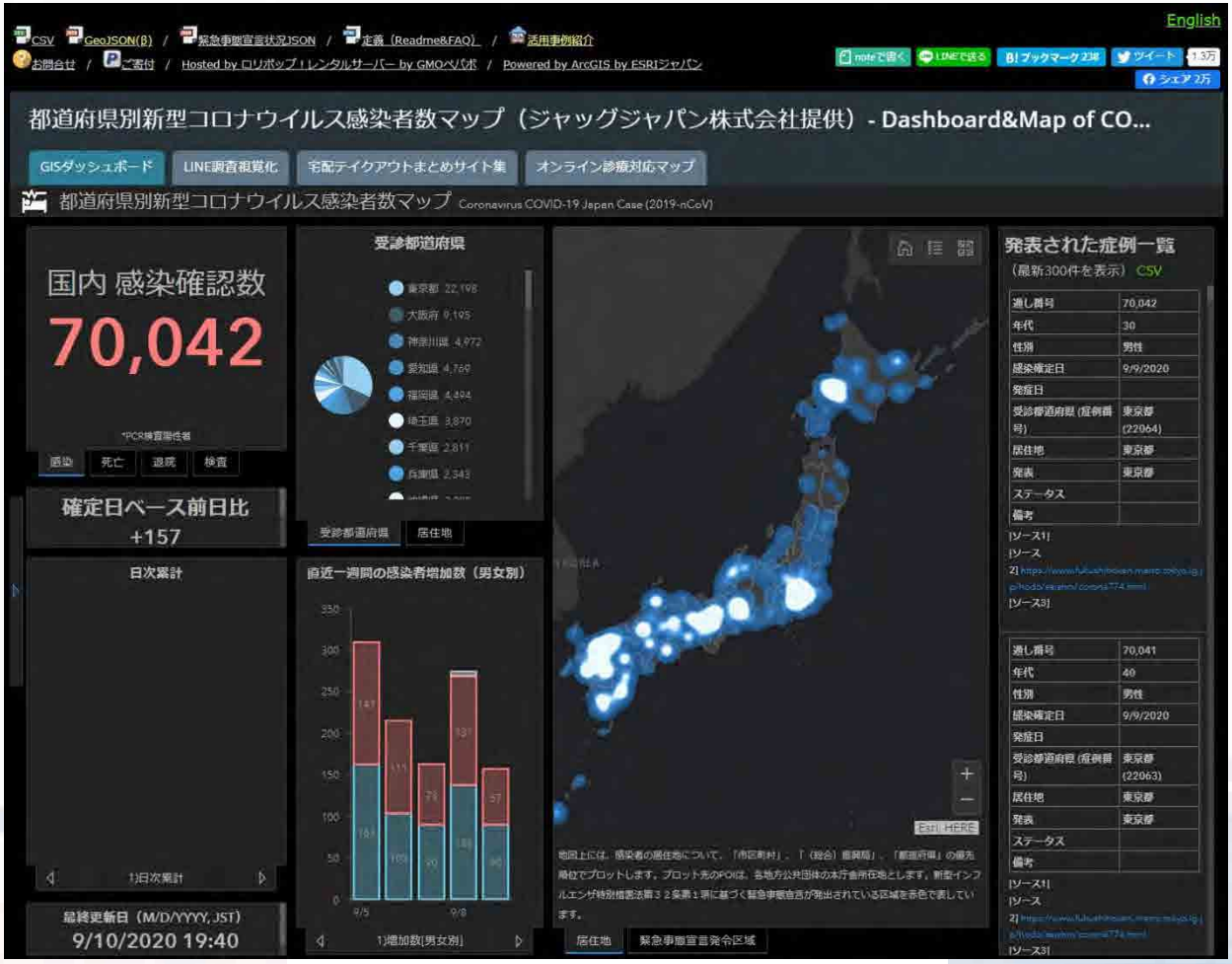
2021/02/25 現在



https://gis.jag-japan.com/covid19jp/ より

都道府県別新型コロナウィルス感染者数

2020/10/01 現在



https://gis.jag-japan.com/covid19jp/ より

都道府県別新型コロナウィルス感染者数

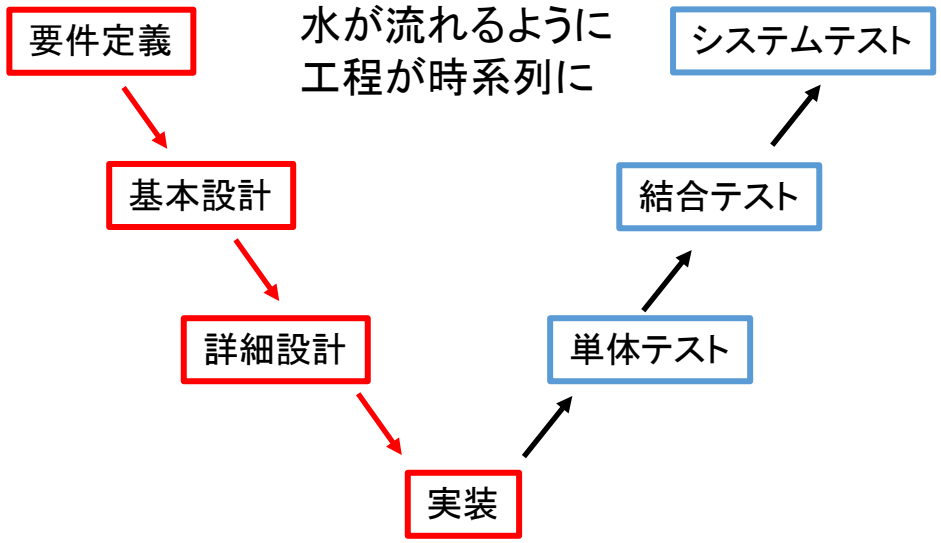
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	通し	厚労省NO	無症状病原体保有者	国内	チャーター便	年	性別	確定日	発症日	受診都道府県	居住都道府県	居住	居住市区町村	キー	発表
2	1	1		A-1		30	男性	1/15/2020	1/3/2020	神奈川県	神奈川県			神奈川県	神奈川県
3	2	2		A-2		40	男性	1/24/2020	1/14/2020	東京都	中華人民共和国			中華人民共和国	東京都
4	3	3		A-3		30	女性	1/25/2020	1/21/2020	東京都	中華人民共和国			中華人民共和国	東京都
5	4	4		A-4		40	男性	1/26/2020	1/23/2020	愛知県	中華人民共和国			中華人民共和国	愛知県
6	5	5		A-5		40	男性	1/28/2020	1/22/2020	愛知県	中華人民共和国			中華人民共和国	愛知県
7	6	6		A-6		60	男性	1/28/2020	1/14/2020	奈良県	奈良県			奈良県	厚生労働省
8	7	7		A-7		40	女性	1/28/2020	1/26/2020	北海道	中華人民共和国			中華人民共和国	北海道
9	8	8		A-8		40	女性	1/29/2020	1/20/2020	大阪府	大阪府			大阪府	大阪府
10	9	9			B-1	50	男性	1/30/2020	1/29/2020	不明	中華人民共和国			中華人民共和国	厚生労働省
11	10 -		チャーター無症状2			50	女性	1/30/2020		千葉県	中華人民共和国			中華人民共和国	千葉県
12	11	10		A-9		50	男性	1/30/2020	1/25/2020	三重県	三重県			三重県	厚生労働省
13	12	11		A-10		30	女性	1/30/2020	1/24/2020	東京都	中華人民共和国			中華人民共和国	東京都
14	13	12		A-11		20	女性	1/30/2020	1/28/2020	京都府	京都府		京都市	京都府京都市	京都市
15	14	16			B-4	40	男性	1/30/2020	2/1/2020	千葉県	中華人民共和国			中華人民共和国	千葉県
16	15	13		A-12		20	女性	1/31/2020	1/20/2020	千葉県	千葉県		千葉市	千葉県千葉市	千葉県
17	16 -		チャーター無症状3			30	男性	1/31/2020		不明	不明			不明	厚生労働省
18	17	14			B-2	40	男性	2/1/2020	1/31/2020	不明	不明			不明	厚生労働省
19	18	15			B-3	40	男性	2/1/2020	1/26/2020	埼玉県	中華人民共和国			中華人民共和国	埼玉県
20	19 -		チャーター無症状5			30	男性	2/1/2020		不明	不明			不明	厚生労働省
21	20	17		A-13		30	女性	2/4/2020	1/30/2020	千葉県	中華人民共和国			中華人民共和国	千葉県
22	21	18			B-5	50	女性	2/4/2020	1/31/2020	不明	中華人民共和国			中華人民共和国	厚生労働省
23	22	19		A-14		50	男性	2/4/2020	1/26/2020	不明	中華人民共和国			中華人民共和国	厚生労働省
24	23	21		A-16		20	男性	2/4/2020	1/25/2020	京都府	京都府		京都市	京都府京都市	京都市
25	24	20		A-15		40	男性	2/5/2020	1/24/2020	千葉県	中華人民共和国			中華人民共和国	千葉県
26	25	22			B-6	50	男性	2/5/2020		不明	中華人民共和国			中華人民共和国	厚生労働省
27	26	23			B-7	20	男性	2/7/2020	2/7/2020	不明	中華人民共和国			中華人民共和国	厚生労働省
28	27	24			B-8	40	男性	2/10/2020	2/8/2020	埼玉県	埼玉県			埼玉県	埼玉県
29	28	25			B-9	50	男性	2/10/2020	2/7/2020	千葉県	中華人民共和国			中華人民共和国	千葉県
30	29	26		A-17		50	男性	2/11/2020		神奈川県	神奈川県			神奈川県	神奈川県

開発

- どのような手順で開発しているか
  - ウォーターフォール開発
  - アジャイル開発
- どのような問題があるか
- 研究で DB を利用する場合によく起こる問題

従来型の開発手法

ウォーターフォール（Water Fall）開発



要件定義：
　要求を実現させるために
　必要な機能や性能を定義

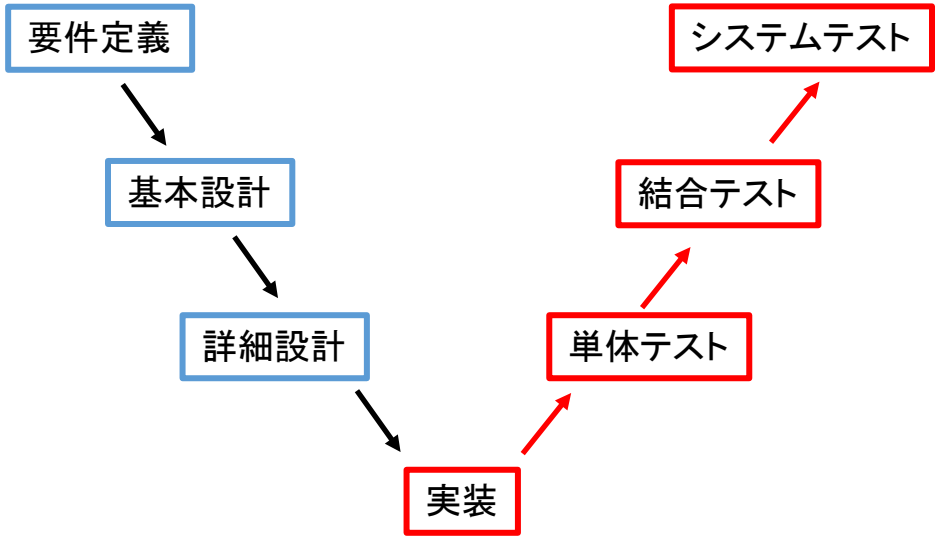
基本設計：
　ユーザーインターフェース等

詳細設計：
　ソフトウエア内部構造の決定

実装：
　実工程、プログラミング

従来型の開発手法

ウォーターフォール（Water Fall）開発



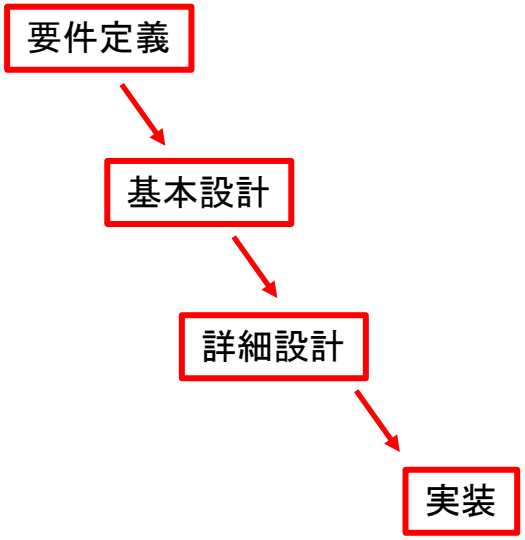
システムテスト：
　システム全体が矛盾なく動作するかをテスト

結合テスト：
　各部を結合させテスト

単体テスト：
　モジュール各部分のテスト

実装：
　ドキュメント通りに作成

ウォーターフォール（Water Fall）開発の特徴・問題



開発をいくつかの工程に分けて順番に取り組んでいくため工程管理がしやすい

各工程でドキュメントが作成され、役割分担が明確化

実物を見て初めて必要な機能やアイデアに気づくことが多い

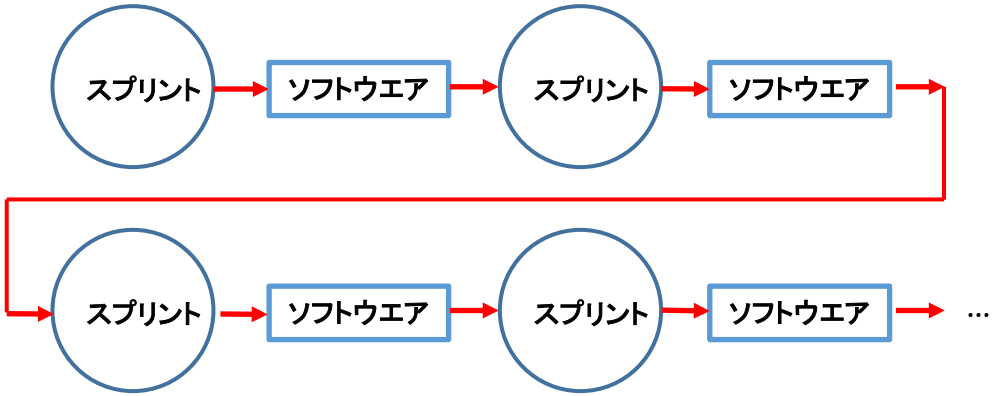
前工程への後戻りが原則的にできない

フィードバックループが生まれない

新しい開発手法

アジャイル（Agile）開発

開発が反復的に進む



スプリント：
　一定の開発工期

ソフトウエア：
　動作するプログラムの作成

新しいソフトウエアが作られ、そこから得られた気づきを基にして新しく要件定義の見直しを行っていくフィードバックループが生まれる.

開発手法まとめ

- ウォーターフォール開発
  - 工程が明確に分離され、役割分担がしやすい.
  - 納期、見積もりが明確になる. 各工程で詳細なドキュメント作成.
  - 原則的に後戻りできない. 後になって必要な機能、不要な機能が出る.
  - 訴訟問題に発展した場合の証拠となる

- アジャイル開発
  - 開発が反復的に進むため、必要な機能が実現されやすい.
  - 開発に不確実性が多い. 見積もりがしづらい.
  - 工程管理、開発チームの管理が難しい.

- 何れの開発方法にも一長一短があり、あらゆる面で有利な開発方法はない

  研究分野における利用では新たな機能が必要になることが多い.

  研究者が DBMS を直接利用する場面が発生.

データベース言語 SQL について

- 1986 年に ANSI によって標準化
- SQL は何かの略語ではない
- 制御文法の仕様がないため、宣言型プログラミング言語と呼ばれる
  - ※ 宣言型プログラムでは達成すべき目的（出力）を示して、それを実現する手続きはシステムに任せる

- 手続き的記述を可能にしたストアードプロシージャーはデータベース管理システム（DBMS）によって仕様が異なり、互換性がない
- DBMS には Oracle, MySQL, PostgreSQL, MariaDB, SQLite などがある

- 今回の演習では互換性のない SQL 命令は基本的に扱わない

SQL コマンド

ひとまずSQL を使ってみる

SQLの命令体系　4大命令

SELECT [列名]　**FROM** [テーブル名]　　**（WHERE　修飾）**;

UPDATE　　[テーブル名] **SET** [列名]　　**（WHERE　修飾）**;

DELETE　　**FROM** [テーブル名]　　**（WHERE　修飾）**;

INSERT　　**INTO** [テーブル名] **VALUES (..)**　　**（WHERE　修飾）**;

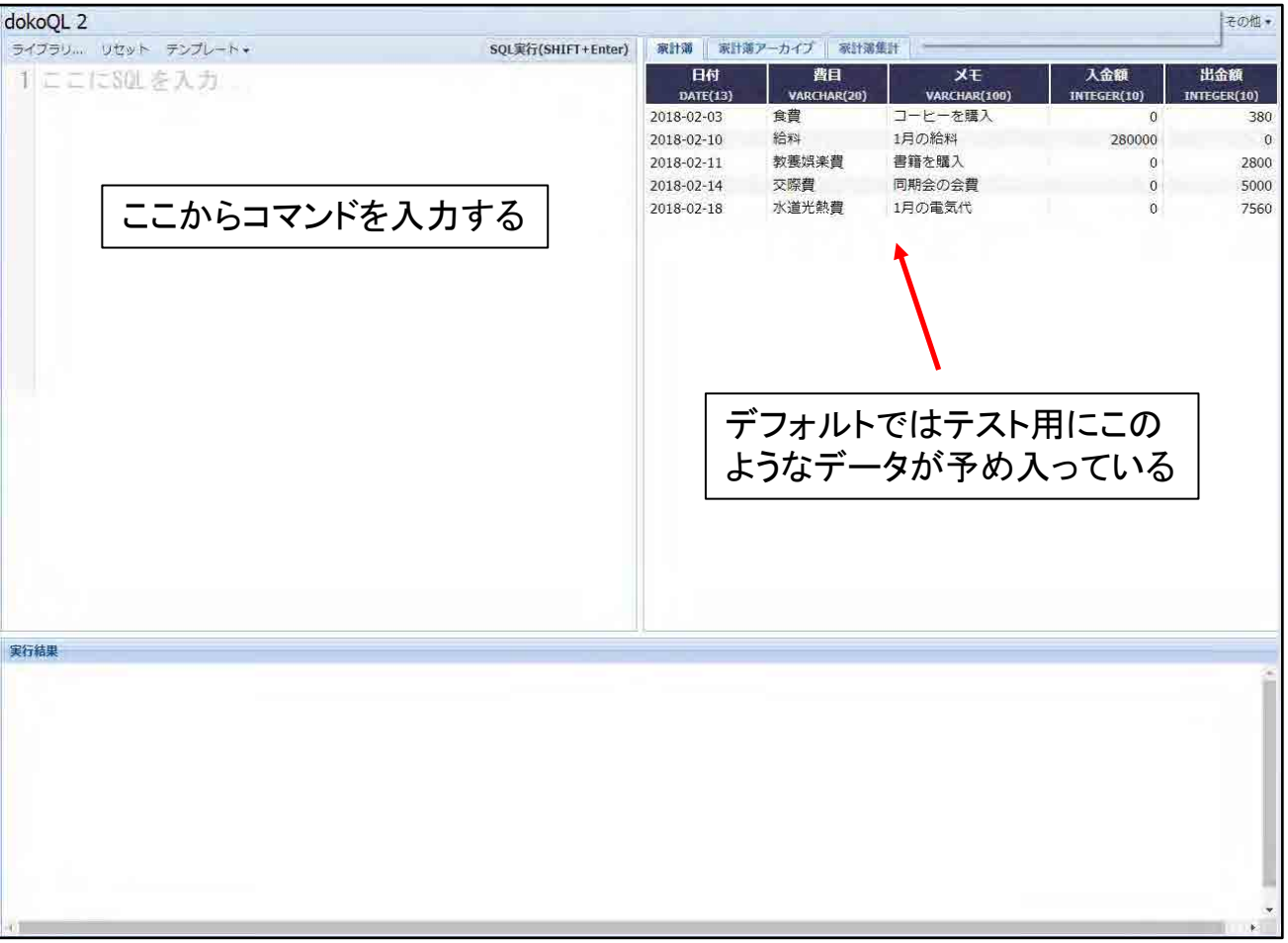
SQLを使ってみる

- dokoQL
  - https://dokoQL.com/

    Web 上から試用が可能な簡易DBMS

- A5SQL（SQL開発ツール）
  - https://a5m2.mmatsubara.com/より

    a5m2_2.15.1_x64.zip or a5m2_2.15.1_x86.zip をダウンロード



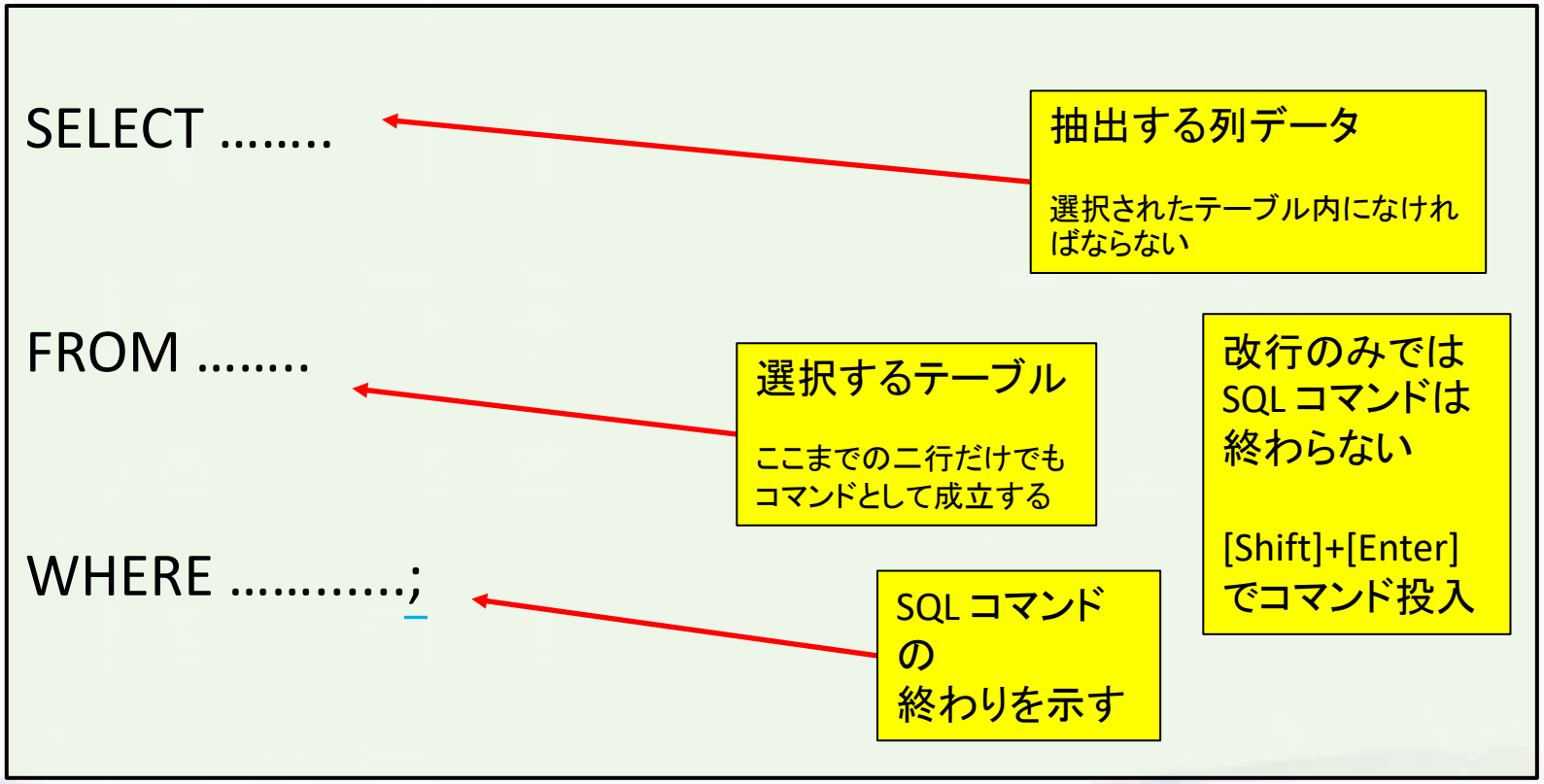
ここからコマンドを入力する

デフォルトではテスト用にこのようなデータが予め入っている

「家計簿」テーブルが既に作成されているので、これをアクセスする

間違えて消去してもリロードすれば最初に戻るため、初期の学習用としてとても便利

SQL の命令体系　SELECT



（1）

```
SELECT * FROM 家計簿;
```

すべての列を選択

家計簿テーブルの中から選択

SQL コマンドの終わりを示す

（2）

```
SELECT 日付, 費目, メモ FROM 家計簿;
```

三項目だけが抽出される

（3）

```
SELECT 日付, 費目, メモ, 入金額
FROM 家計簿
WHERE 入金額 > 0;
```

改行のみではSQL コマンドは終わらない

入金額が 0 以上のものだけが抽出される

（4）

```
SELECT  日付, 費目, メモ, 出金額
FROM 家計簿
WHERE 日付 >= '2018-02-05';
```

シングルクォートで括った文字は文字列と認識される。
特定の形式で入力されている場合は（ここでは日付）その形式に対応する情報として認識される。

この日以降のデータを表示する

（5）

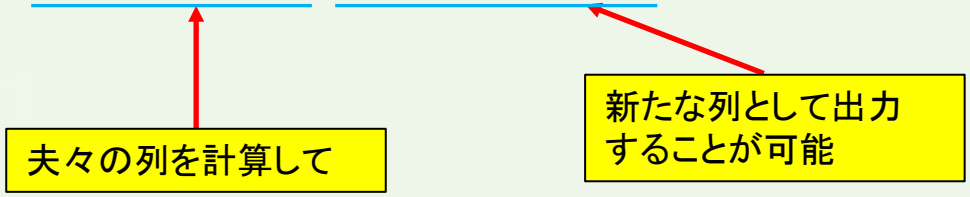
```
SELECT  日付, 費目, メモ, 出金額
FROM 家計簿
WHERE 出金額 > 0
ORDER BY 出金額 DESC;
```

降順で表示する

昇順なら ASC を用いる

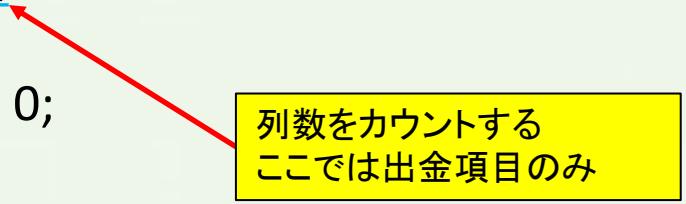
（6）

```
SELECT  日付, 費目, メモ, 出金額*1.10 AS 税込出金額
FROM 家計簿;
```

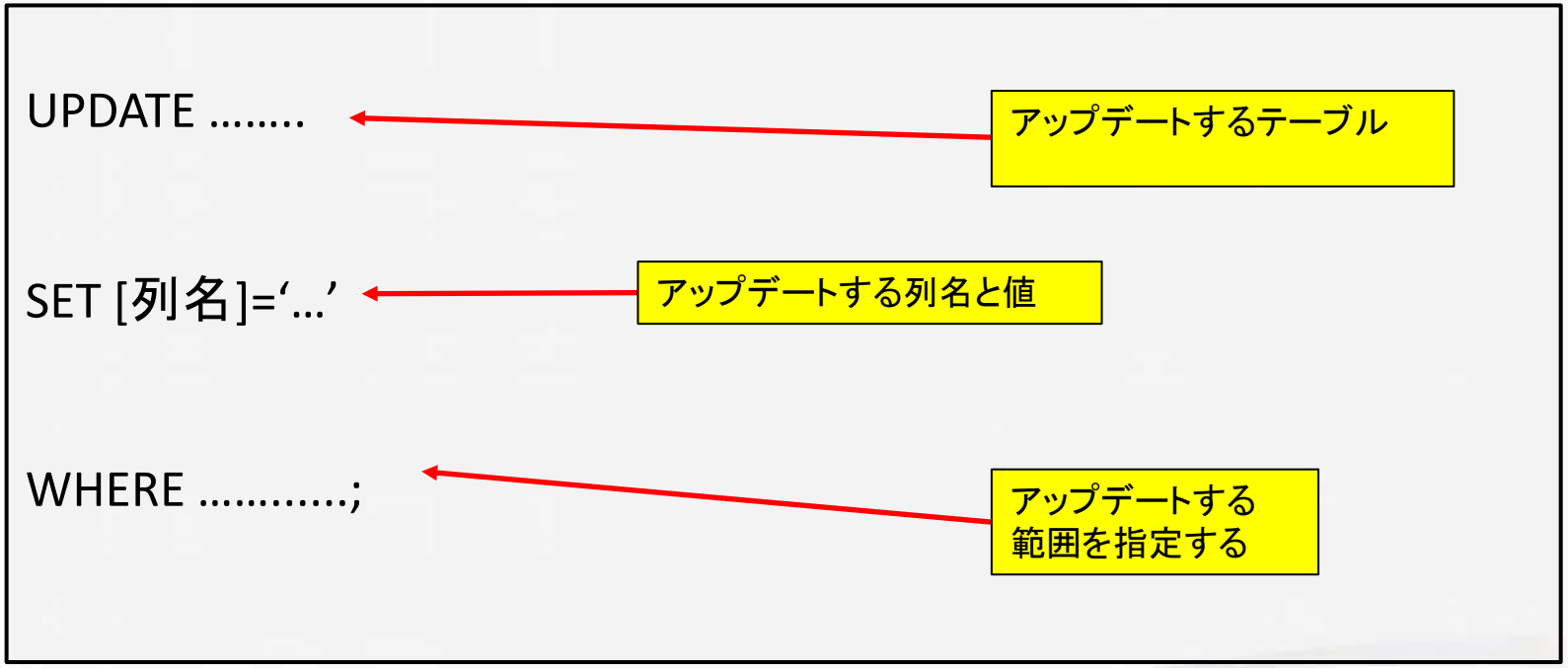


（7）

```
SELECT COUNT(*) AS 出金費目数
FROM 家計簿
WHERE 出金額 > 0;
```



SQLの命令体系　UPDATE



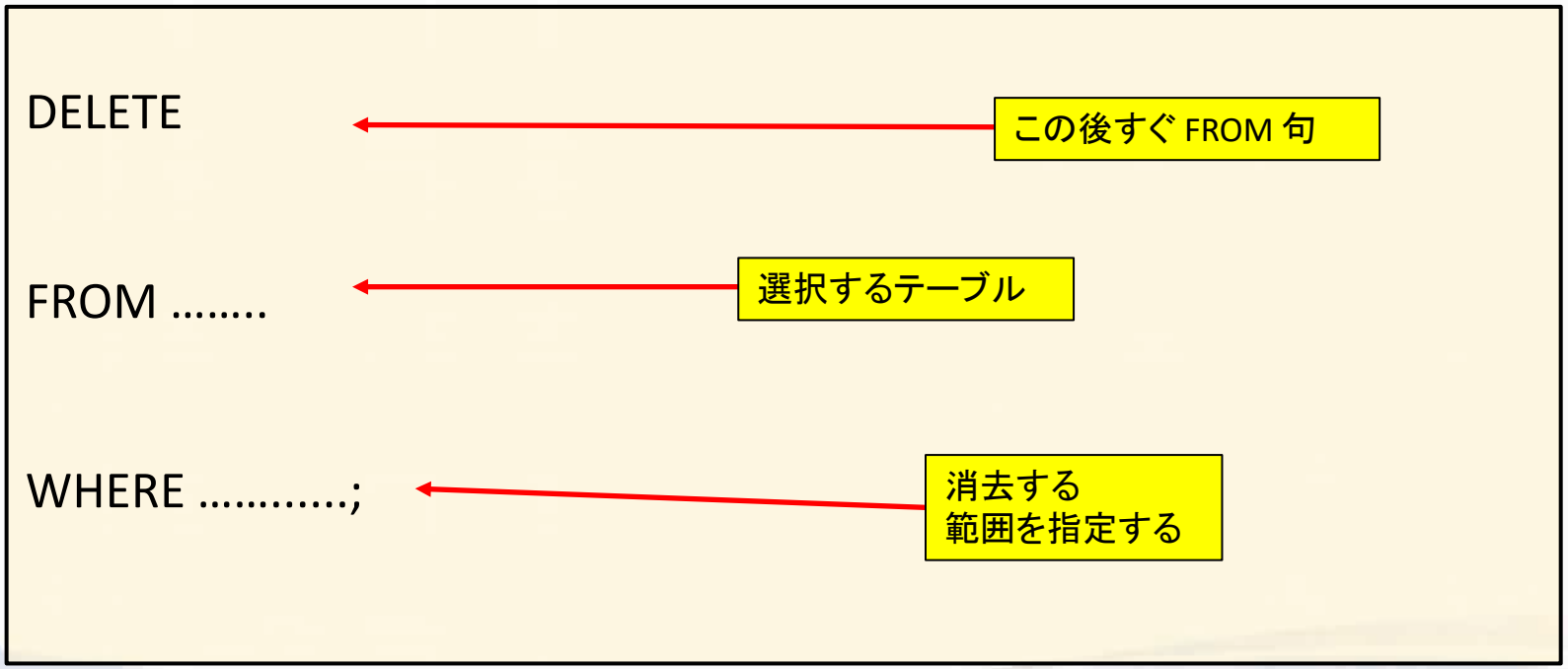
SQLの命令体系　UPDATE

（1）
UPDATE 家計簿 SET 入金額 = 9999;

（2）
UPDATE 家計簿 SET 入金額 = 9999 WHERE 日付 =‘2018-02-10’;

（問）どの場所が書き換わるか、予め調べる方法は？

SQL の命令体系　DELETE



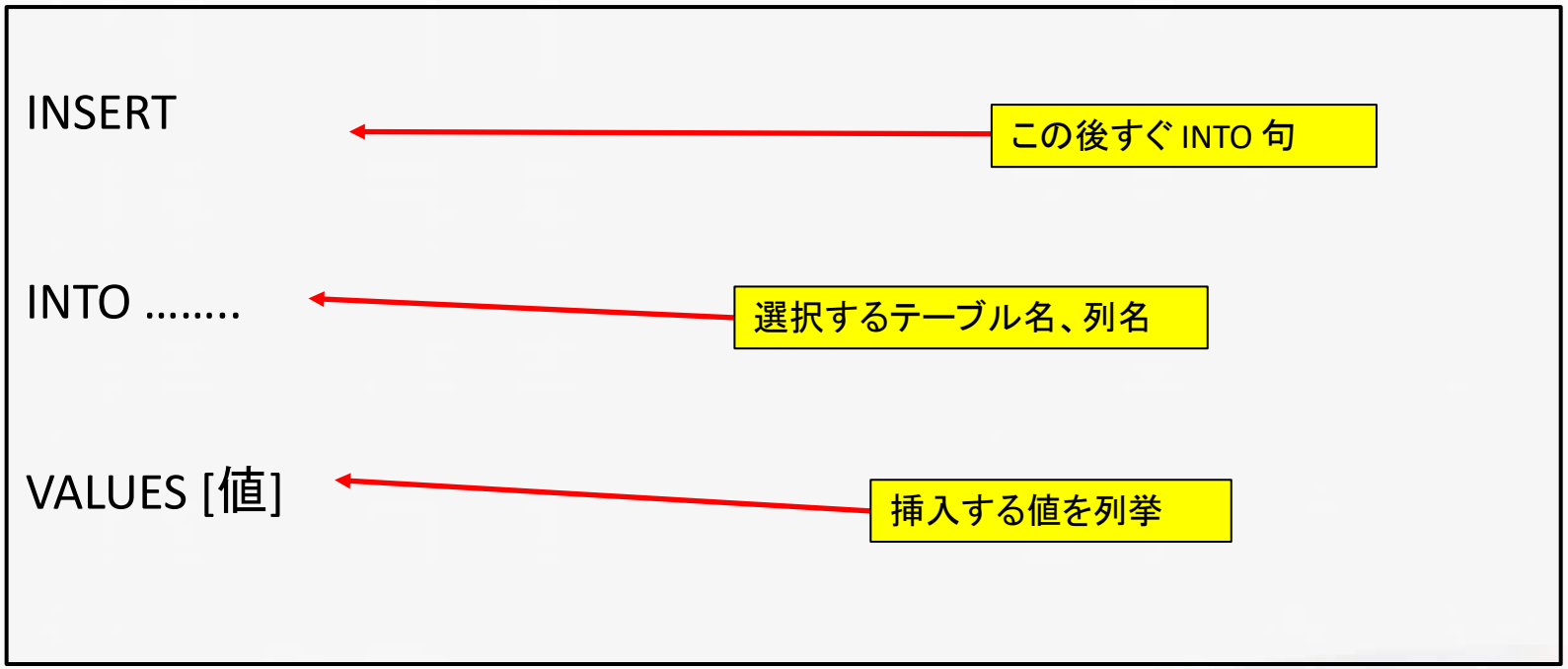
SQL の命令体系　DELETE

（1）
DELETE FROM 家計簿 ;

（2）
DELETE FROM 家計簿 WHERE 入金額 = 0;

（問）
出金額が 4000 円以下の行を消去せよ

SQL の命令体系　INSERT



SQLの命令体系　INSERT

（1）
INSERT INTO 家計簿 (費目,日付,出金額)
VALUES ('交通費','2018-02-25',1200);

（2）
INSERT INTO 家計簿
VALUES ('交通費','2018-02-25',1200);

（3）
INSERT INTO 家計簿
VALUES ('2018-02-25', '交通費', '東京メトロ', 0,1200);

（問）NULL 行があるかどうか調べる方法は？

SQL の命令体系　比較演算（1）

NULL の使い方

[列名]　IS NULL　[列名] IS NOT NULL などと表記

[列名] = NULL　　[列名] <> NULL とは書かない

SQLの命令体系　比較演算(2)

数値の比較の方法

=　左右の値が等しい

<　左辺が右辺より小さい
<=　左辺が右辺より小さい、または等しい

>　右辺が左辺より小さい
>=　右辺が左辺より小さい、または等しい

<>　左右の値が等しくない

SQLの命令体系　比較演算（3）

LIKE の使い方

LIKE '%[文字列]'　前方一致

LIKE '%[文字列] %'　　[文字列]を含む

LIKE '_[文字列]'　[文字列]の前に任意の１文字を含む

SQL の命令体系　比較演算（4）

IN の使い方

WHERE 費目 IN ('食費', '交際費');

WHERE 費目 NOT IN ('食費', '交際費');

（問）
NOT IN を用いた SQL 文を作成せよ

SQL の命令体系　比較演算（5）

AND　OR　の使い方

WHERE 日付 > '2018-02-10' AND 出金額 < 5000;

WHERE 日付 > '2018-02-10' AND 出金額 > 5000 OR 入金額 > 5000;

AND の方が優先順位が高い

WHERE 日付 > '2018-02-10' AND (出金額 > 5000 OR 入金額 > 5000);

（問）
SQL 文を完成させて確認せよ

データサイエンス応用コース

構造化データ・蓄積・加工（2）

2021/03/04

大阪大学 特任准教授
shimokawa@sigmath.es.osaka-u.ac.jp 下川和郎

データベースを設計する

ex. 人の健康情報に関するデータベース

- 概念設計
  - どのような情報の塊を管理するか
- 論理設計
  - データベースにとって扱いやすい構造を作る　テーブル、主キー
  - 正規化（第一、第二、第三正規形）
- 物理設計
  - どの DBMS を用いるか、OS, ハードウエア、制約、インデックス

概念設計

- 個人情報の塊
  - 住所、氏名、生年月日、性別、電話番号、個人番号、…
- 検体検査情報の塊
  - 調査票等【身長、体重、問診票】
  - 外部検査機関から【血液検査、尿検査】
- 分子情報の塊
  - （ゲノム情報、SNP 情報、遺伝子発現情報等）

概念設計：個人情報

個人 ID

死亡年月日
死亡した場合に入力

ヒアリング等

更新日時　更新者

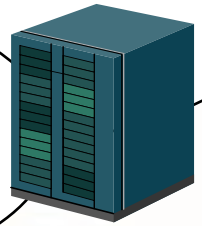
生年月日

姓　名　旧姓

予備

運用は？

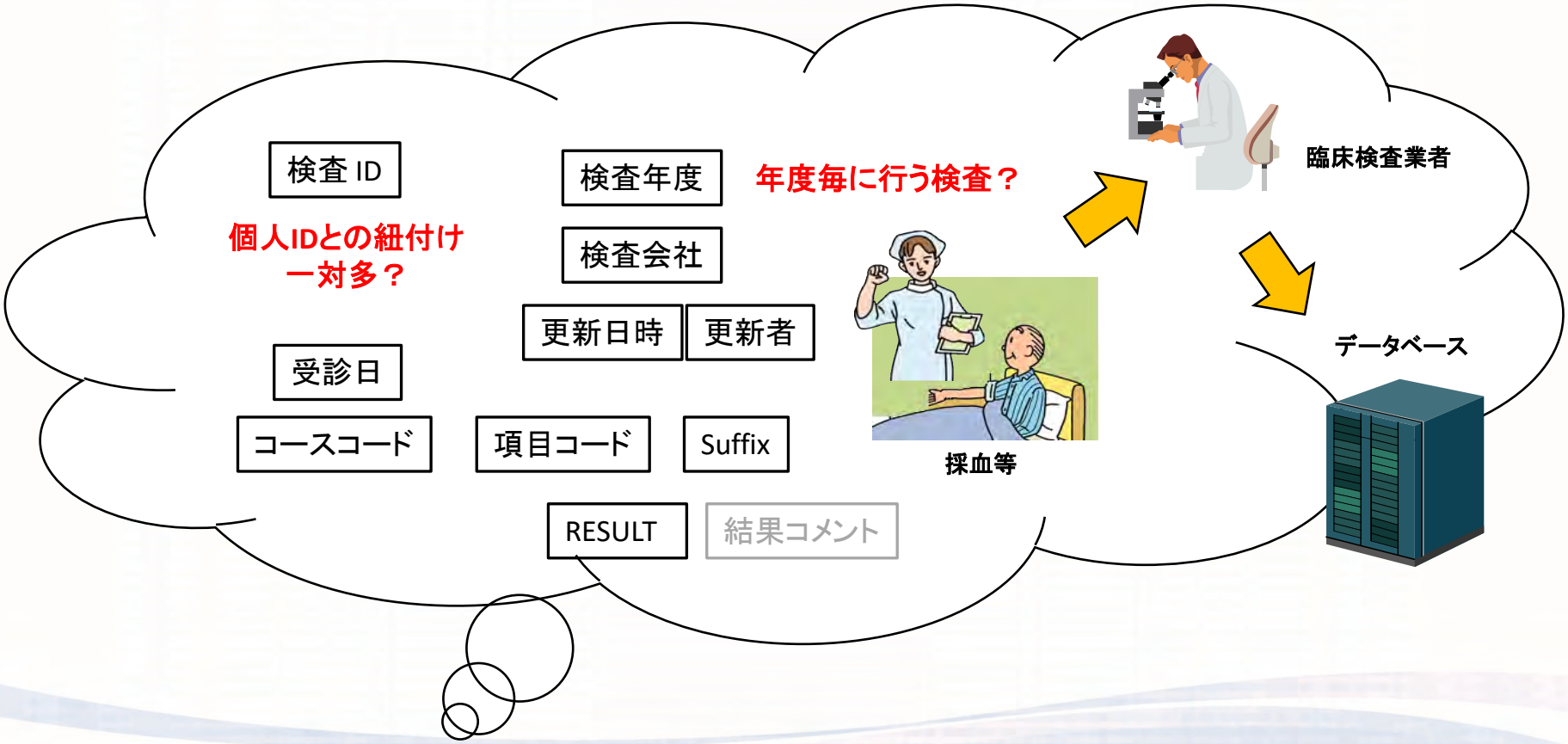
データベース



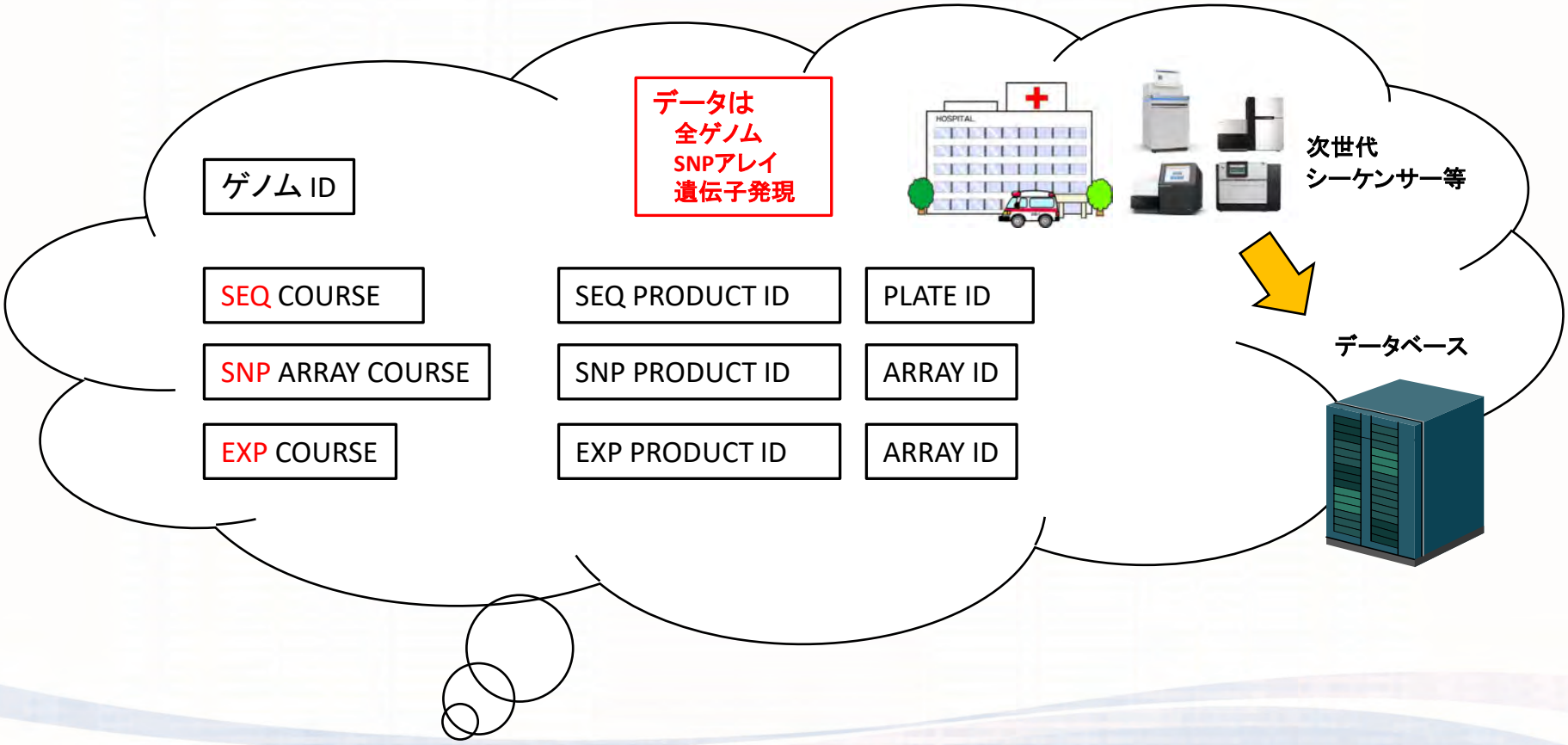
性別
2つでよい？

撤回レベル
データ入力同意を撤回した場合

概念設計：検体検査情報


検査 ID
検査年度
年度毎に行う検査？
個人IDとの紐付け
一対多？
検査会社
更新日時
更新者
受診日
コースコード
項目コード
Suffix
RESULT
結果コメント
採血等
臨床検査業者
データベース

概念設計：分子情報


データは
全ゲノム
SNPアレイ
遺伝子発現
HOSPITAL
次世代
シーケンサー等
ゲノム ID
SEQ COURSE
SEQ PRODUCT ID
PLATE ID
SNP ARRAY COURSE
SNP PRODUCT ID
ARRAY ID
EXP COURSE
EXP PRODUCT ID
ARRAY ID
データベース

論理設計

- 多対多の関係を一対多の関係へ
- キーの整理
  - 主キーを設定する
- テーブルの分割
  - 個人情報テーブル
  - 検体検査情報テーブル
  - 分子情報テーブル
  - 関係情報テーブル

正規化 (第一～第三正規形)

- 第一正規形
  - テーブル内のセルをこれ以上分割できない単位に分割する
- 第二正規形
  - 部分関数従属(主キーをなす複合キーの一つに関数従属する列)が存在する場合、これを切り離して外部テーブルに.
- 第三正規形
  - 推移関数従属(主キーに関数従属する列に、さらに関数従属する列)が存在する場合、これを切り離して外部テーブルに

(主キー:主キーが決まれば他の列が一意に決まる)

(複合主キー:複数の列の組み合わせで他の列が一意に決まる)

(関数従属:列Aが決まれば列Bが一意に決まる)

正規化 (第一正規形)

ex. 検体検査情報テーブル

例:質問項目で複数回答がある場合.

主キー

ITEMCODE	Suffix	JLAC10	RESULT
405050			167.3
405060			59.4
405070			1, 2

複合主キー

複合キーの一部となる

ITEMCODE	Suffix	JLAC10	RESULT
405070	00		1
405070	01		2

複数データは分割

一つのセルには一つのデータを入れる

正規化 (第一正規形)

主キーに対して複数の値が格納されていた.

複合主キーを作成して値を異なる場所へ格納.

正規化 (第二正規形)

ex. 分子情報テーブル

例：個人の遺伝情報を三つの検査方法で取得した.

データ	ID	製品
全ゲノム	00	SEQ
SNPアレイ	01	SNP
遺伝子発現	02	EXP

従属

ID

製品

部分関数従属

GENID	DATATYPE	PRODUCT	PRODUCT_ID	PLATE_ID	…
01000	00	SEQ	S0148	14846	…
01000	01	SNP	N037	3745	…
01001	00	SEQ	S094	9455	…
01002	00	SEQ	S038	3874	…
01002	02	EXP	E047	4734	…
01003	01	SNP	N083	8345	…
01003	02	EXP	E015	1543	…
01004	01	SNP	N085	8564	…
01005	00	SEQ	S012	1245	…

複合主キー

部分関数従属：複合主キー DATATYPE に PRODUCT が従属

正規化 (第二正規形)

ex. 分子情報テーブル

例：個人の遺伝情報を三つの検査方法で取得した.

データ	ID	製品
全ゲノム	00	SEQ
SNPアレイ	01	SNP
遺伝子発現	02	EXP

ID

GENID	DATATYPE	PRODUCT_ID	PLATE_ID	…
01000	00	S0148	14846	…
01000	01	N037	3745	…
01001	00	S094	9455	…
01002	00	S038	3874	…
01002	02	E047	4734	…
01003	01	N083	8345	…
01003	02	E015	1543	…
01004	01	N085	8564	…
01005	00	S012	1245	…

複合主キー

ID 製品

DATATYPE	PRODUCT
00	SEQ
01	SNP
02	EXP

二つのテーブルに分割

正規化 (第二正規形)

複合主キーのうちの一つに対して関数従属する列があった.

別テーブルを作成してその列を異なる場所へ格納.

正規化 (第三正規形)

ex. 検体検査情報テーブル

例：検査会社が度々変更.

従属

検査会社 ID　　検査会社名

RSLID	INSPECTYEAR	INSCMPNID	PLATE_ID	PNAME
194785	2017	00	A01673	A LABO
154857	2018	01	B01467	B RESEARCH
176494	2019	02	C01367	C Institute
184664	2017	00	A01168	A LABO
114758	2018	00	A01367	A LABO
147685	2019	02	C01964	C Institute
153695	2017	00	A01353	A LABO
194755	2018	02	C01866	C Institute
184672	2019	02	C01345	C Institute

主キー

従属

推移関数従属：主キーRSLID に 従属する INSCMPNID にさらに PNAME が従属

設定

検査会社が毎年変更になった

2018 年度のみ複数の検査会社がデータを納入した

2017年　A LABO

2018年　A,B,C 社

2019年　C Institute

正規化 (第三正規形)

ex. 検体検査情報テーブル

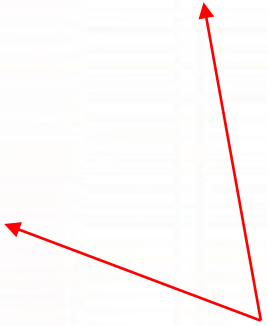
例：検査会社が度々変更.

検査会社 ID

RSLID	INSPECTYEAR	INSCMPNID	PLATE_ID
194785	2017	00	A01673
154857	2018	01	B01467
176494	2019	02	C01367
184664	2017	00	A01168
114758	2018	00	A01367
147685	2019	02	C01964
153695	2017	00	A01353
194755	2018	02	C01866
184672	2019	02	C01345

検査会社 ID　検査会社名

INSCMPNID	PNAME
00	A LABO
01	B RESEARCH
02	C Institute



二つのテーブルに分割

設定

検査会社が毎年変更になった

2018 年度のみ複数の検査会社がデータを納入した

2017年　A LABO

2018年　A,B,C 社

2019年　C Institute

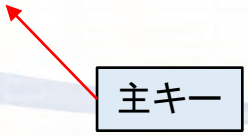
正規化 (第三正規形)

主キーに関数従属する列に対して関数従属する列があった.

別テーブルを作成してその列を異なる場所へ格納.

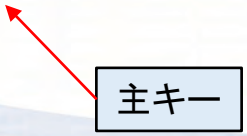
論理設計：個人情報 (Table：PERSON)

PERID	UPDDATE	BIRTH	JENDER	SUR NAME	FIRST NAME	DEATH DATE	DEATHDATE	WITHDRAW	RESERVE
04101	2018-01-03	1985-03-26	M	松崎	清志			0	
04102	2017-01-26	1994-06-20	M	加藤	真			0	
04103	2014-04-14	1971-05-16	F	丘	浩太郎	野村		0	
04104	2018-01-08	1989-08-01	M	黒尾	邦夫			0	
04105	2018-07-04	1994-01-16	F	五十里	前代	宮田		0	
04106	2012-05-06	1978-06-27	M	植野	竜太			2	
04107	2014-12-11	1993-09-04	F	若林	博文	洞口		0	
04108	2014-08-25	1986-11-08	F	生川	博			1	
04109	2016-03-06	1980-05-23	M	後藤	伸一			0	



論理設計：検体検査情報 (Table：RSLTEST)

RSLID	RSLDATE	INSCMPNID	COURSE	ITEMCODE	SUFFIX	JLAC10	RESULT	COMMENT	…
194785	2013/3/2	00	1000	405050	00		162.3		…
154857	2013/3/2	01	1000	405060	00		59.4		…
176494	2013/3/2	02	1000	405070	00		1		…
184664	2014/7/14	00	2000	405050	00		167.3		…
114758	2014/7/14	00	2000	405060	00		63.8		…
147685	2014/7/14	02	2000	405070	00		2		…
153695	2013/7/19	00	1000	405050	00		172.1		…
194755	2015/3/2	02	…	…	…	…	…	…	…
184672	2015/3/2	02							



論理設計：分子情報 （Table：GENOMETBL)

GENID	DATATYPE	PRODUCT_ID	PLATE_ID	…
01000	00	S0148	14846	…
01001	01	N037	3745	…
01002	00	S094	9455	…
01003	00	S038	3874	…
01004	02	E047	4734	…
01005	01	N083	8345	…
01006	02	E015	1543	…
01007	01	N085	8564	…
01008	00	S012	1245	…

論理設計：テーブル間を接続

個人ID　ゲノム ID

PERID	GENID
04101	01000
04102	01001
04103	01002
04104	01003
04105	01004
04106	01005
04107	01006
04108	01007
04109	01008

個人ID　検査結果 ID

PERID	RSLID
04101	194785
04102	154857
04103	176494
04103	184664
04105	114758
04102	147685
04107	153695
04101	194755
04101	184672
…	…

このテーブルはセキュリティー上重要になる

物理設計

- どの OS を用いるか
  - Linux であれば堅牢で高速（利用できない DBMS もある）
  - Windows であれば少ないスキルでメンテナンスが可能
  - ファイルシステムは OS と切り離して別途用意が可能
- ここでは実習のため WindowsノートPC 上で動作することが前提
  また小規模で軽量なデータを扱うため DBMS には SQLite を用いる
- DB アクセスのための GUI は A5SQL を用いる

データサイエンス応用コース

構造化データ･テーブル結合

2021/03/11

大阪大学 特任准教授
shimokawa@sigmath.es.osaka-u.ac.jp 下川和郎

講義の概要

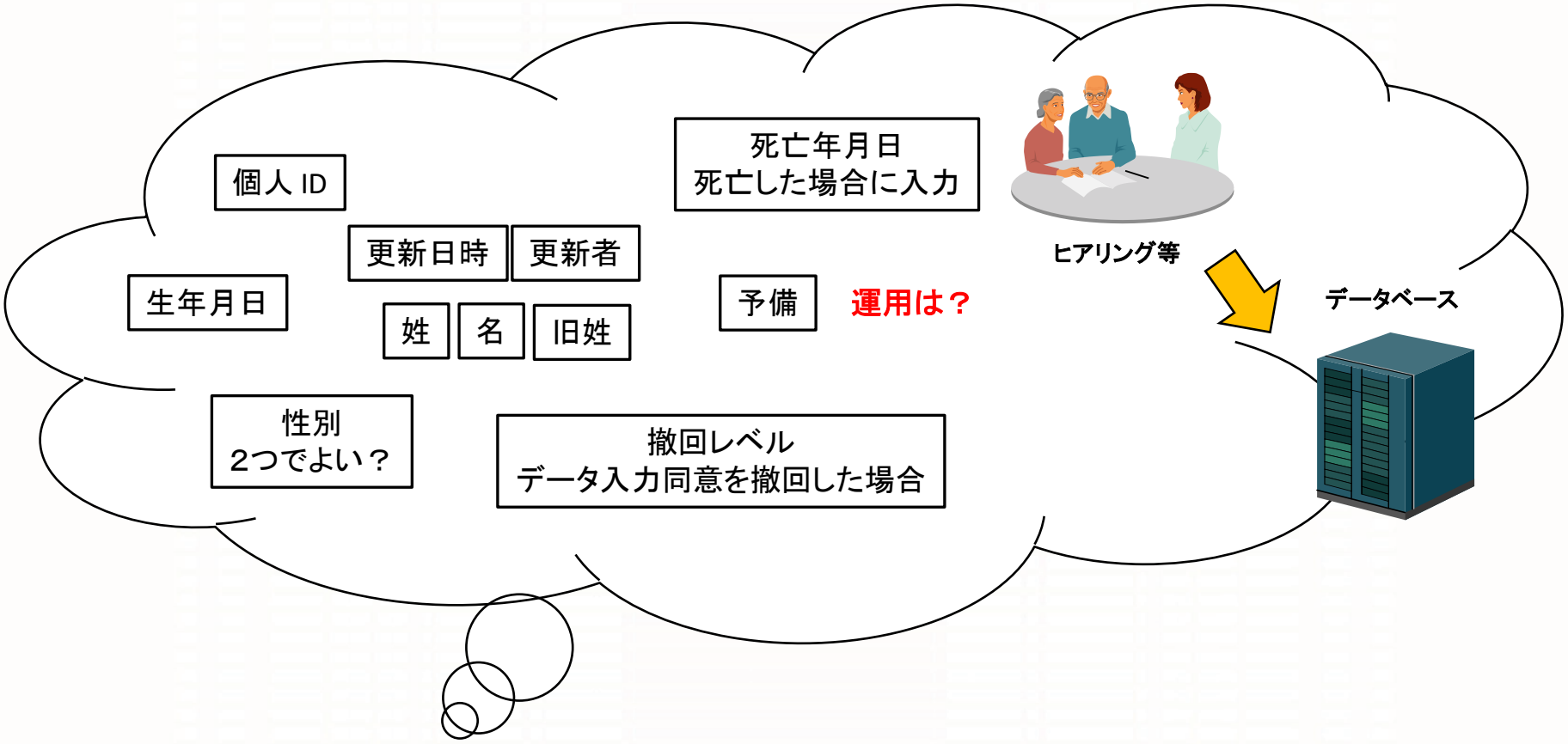
- データベース設計
  - SQL コマンド応用
    - テーブル結合、副問い合わせ
    - データ入力インデックス作成
- 実行制御・セキュリティー
    - トランザクション、ロールバック、デッドロック
    - SQL インジェクション

SQL のデータ型　各カラムで指定

データ種別	区分	代表的なデータ型名	Oracle	SQLite3
数値	整数	INTEGER	NUMBER	INTEGER
	浮動小数点数	DECIMAL, REAL	BINARY_FLOAT …	REAL
文字列	固定長	CHAR	CHAR	TEXT
	可変長	VARCHAR	VARCHAR2	
日付時刻		DATETIME, DATE, TIME	DATE	DATE
その他			NULL BLOB…	NULL BLOB

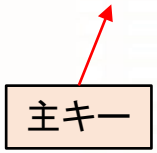
SQLite3 はデータ型制約が甘い.
型指定も必須ではない.
重要な概念だが今回はあまり触れない.

概念設計：個人情報


個人 ID
死亡年月日
死亡した場合に入力
更新日時
更新者
生年月日
姓
名
旧姓
予備
運用は？
ヒアリング等
データベース
性別
2つでよい？
撤回レベル
データ入力同意を撤回した場合

個人情報 (Table：PERSON)

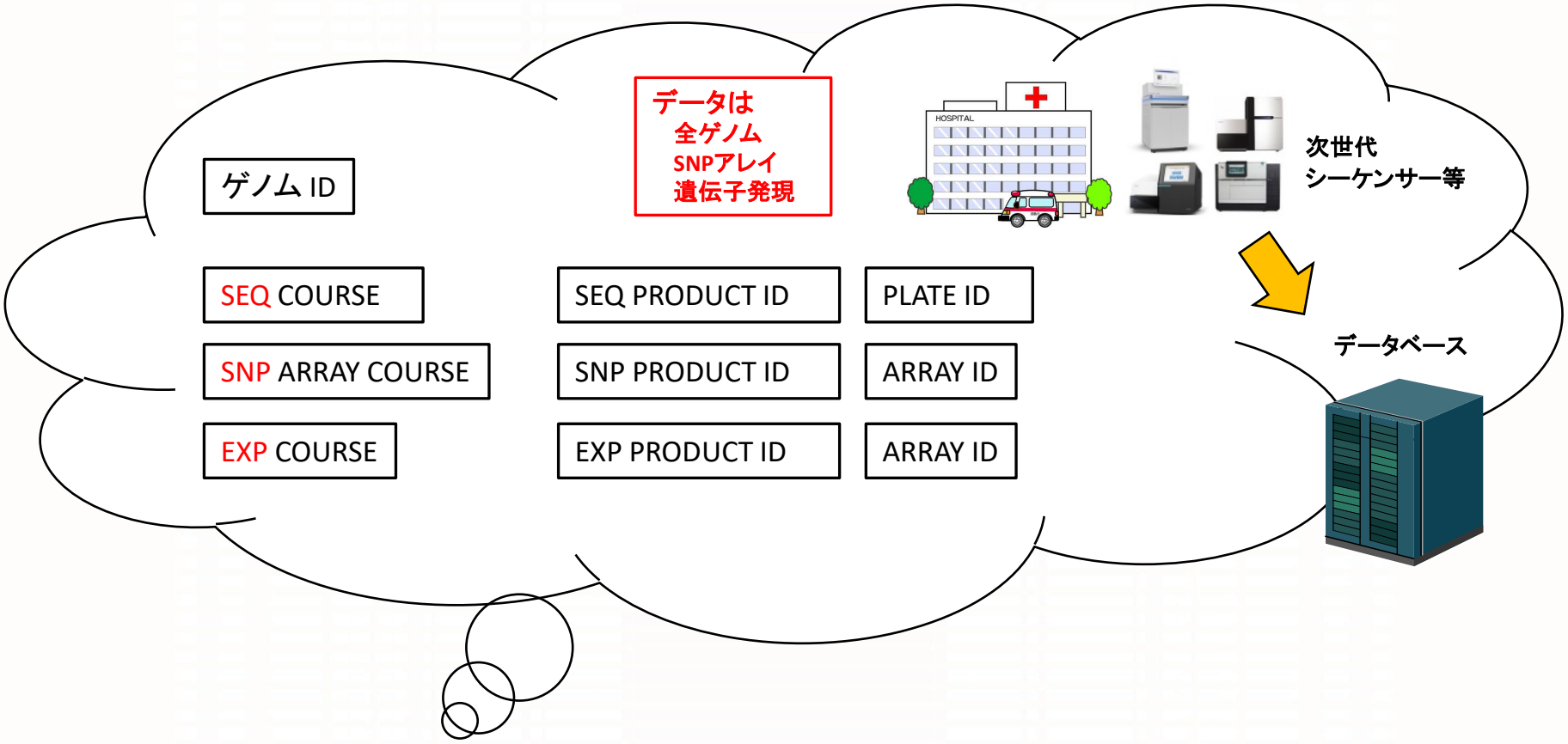
PERID	BIRTH	GENDER	LAST NAME	FIRST NAME	DEATHDATE	WITHDRAW
410001	1985-03-26	1	松崎	清志		0
410002	1994-06-20	1	加藤	真		0
410003	1971-05-16	1	丘	浩太郎		0
410004	1989-08-01	1	黒尾	邦夫		0
410005	1994-01-16	2	五十里	前代		0
410006	1978-06-27	1	植野	竜太		2
410007	1993-09-04	1	若林	博文	2018-07-04	2
410008	1986-11-08	1	生川	博		1
410009	1980-05-23	1	後藤	伸一		0



テーブル作成 (Table: PERSON)

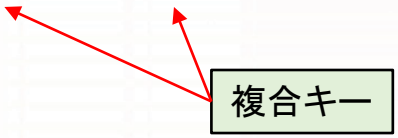
```
CREATE  TABLE  PERSON (
                    PERID TEXT,
                    BIRTH DATE,
                    GENDER  INTEGER(1),
                    LASTNAME TEXT(25),
                    FIRSTNAME TEXT(25),
                    DEATH_DATE DATE,
                    WITHDRAW_LEVEL INTEGER (1)
);
```

概念設計：分子情報


データは
全ゲノム
SNPアレイ
遺伝子発現
HOSPITAL
次世代
シーケンサー等
ゲノム ID
SEQ COURSE
SEQ PRODUCT ID
PLATE ID
SNP ARRAY COURSE
SNP PRODUCT ID
ARRAY ID
EXP COURSE
EXP PRODUCT ID
ARRAY ID
データベース

論理設計：分子情報 (Table：GENOME_TBL)

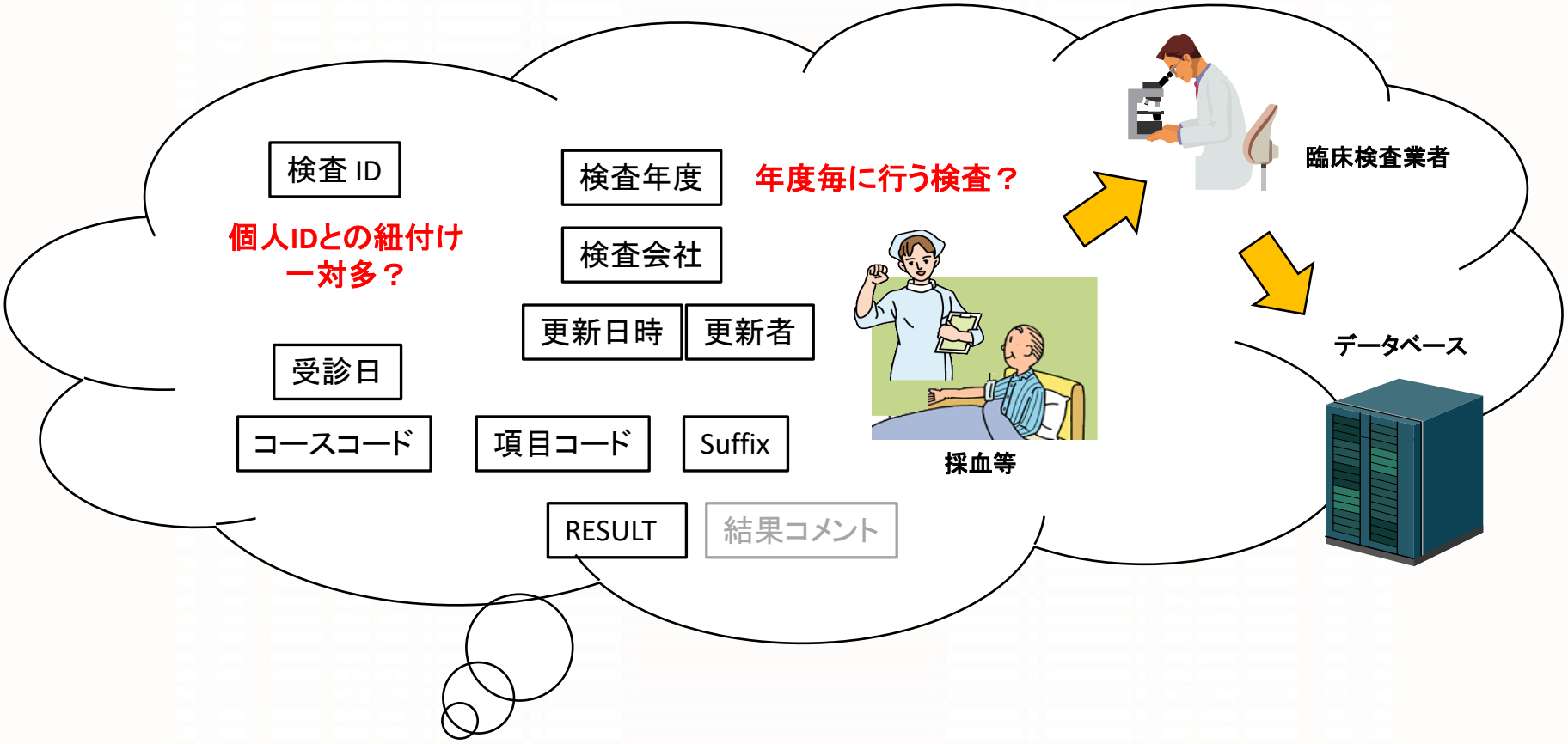
GENID	DATATYPE	PRODUCT_ID	PLATE_ID
01000	0	S0148	14846
01000	1	N037	3745
01001	0	S094	9455
01002	0	S038	3874
01003	2	E047	4734
01004	1	N083	8345
01004	2	E015	1543
01005	1	N085	8564
01006	0	S012	1245



テーブル作成 (Table: GENOME_TBL)

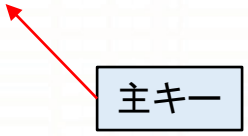
```
CREATE  TABLE  GENOME_TBL (
                    GENID INTEGER(6),
                    DATATYPE INTEGER(1),
                    PRODUCT_ID TEXT,
                    PLATE_ID  TEXT
);
```

概念設計：検体検査情報


検査 ID
個人IDとの紐付け
一対多？
検査年度
年度毎に行う検査？
検査会社
更新日時
更新者
受診日
コースコード
項目コード
Suffix
RESULT
結果コメント
採血等
臨床検査業者
データベース

論理設計：検体検査情報 (Table：RESULT_TBL)

RSLID	RSLDATE	INSCOMP_ID	COURSE	ITEMCODE	SUFFIX	RESULT
194785	2013-3-2	00	1000	405050	00	162.3
154857	2013-3-2	01	1000	405060	00	59.4
176494	2013-3-2	02	1000	405070	00	1
184664	2014-7-14	00	2000	405050	00	167.3
114758	2014-7-14	00	2000	405060	00	63.8
147685	2014-7-14	02	2000	405070	00	2
153695	2013-7-19	00	1000	405050	00	172.1
194755	2015-3-2	02	…	…	…	…
184672	2015-3-2	02				



テーブル作成　(Table: RESULT_TBL)

```
CREATE  TABLE  RESULT_TBL (

                    RSLID  INTEGER(6),
                    RSLDATE  DATE,
                    INSCOMP_ID  INTEGER(2),
                    COURSE  INTEGER(4),
                    ITEMCODE  INTEGER(6),
                    SUFFIX  INTEGER (2),
                    RESULT  TEXT(50)
);
```

テーブル間を結合するために必要な情報

PER_GEN

PERID	GENID
04101	01000
04102	01001
04103	01002
04104	01003
04105	01004
04106	01005
04107	01006
04108	01007
04109	01008

RSL_PER

PERID	RSLID
04101	194785
04102	154857
04103	176494
04103	184664
04105	114758
04102	147685
04107	153695
04101	194755
04101	184672
…	…

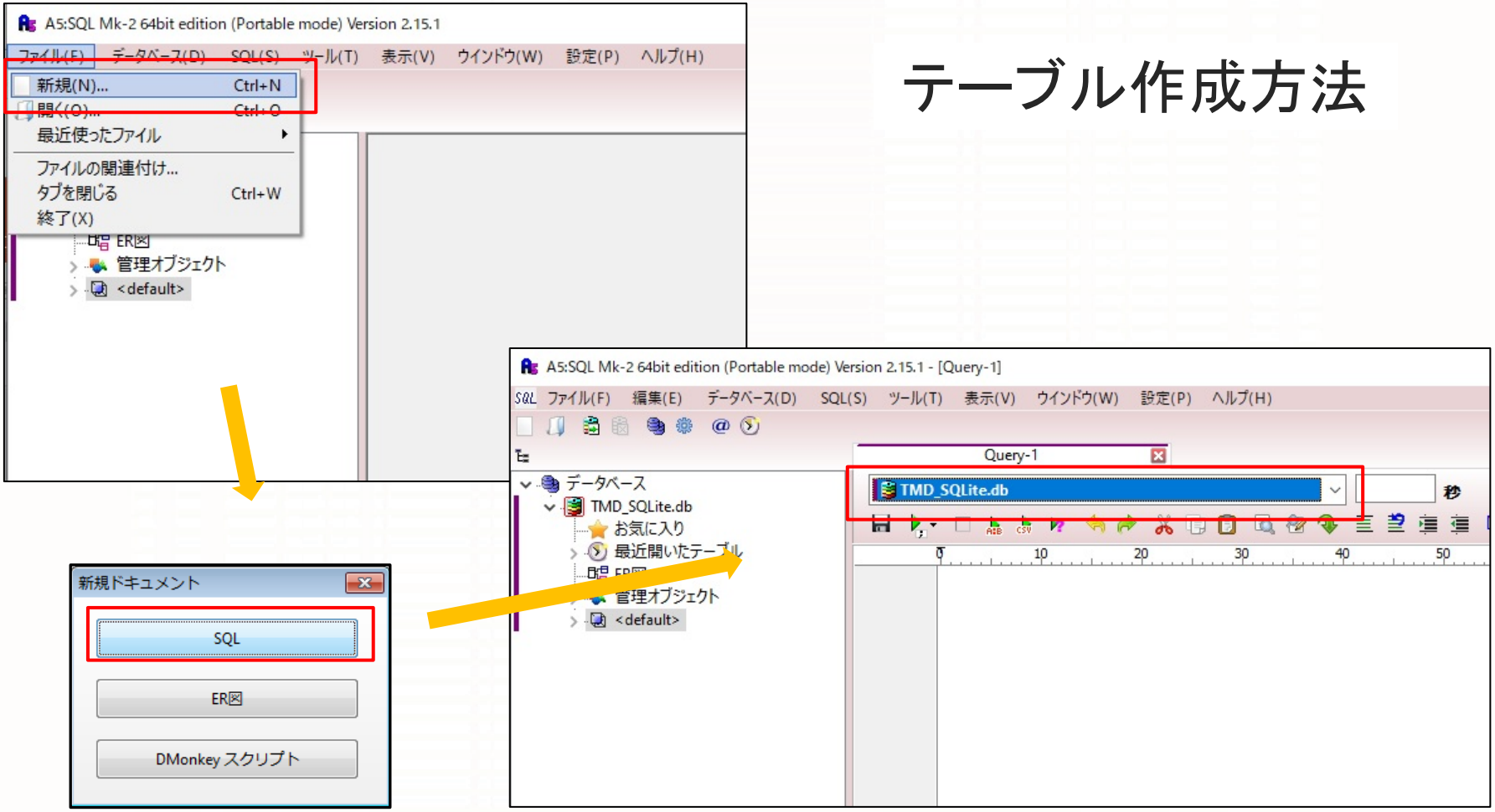
GEN_TYPE

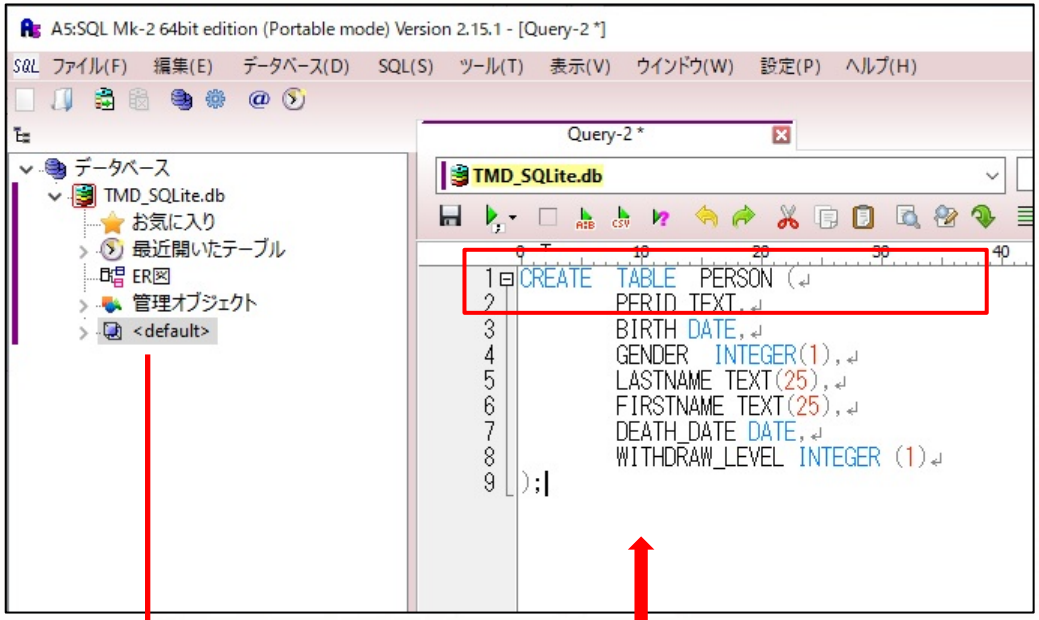
DATATYPE	PRODUCT
0	Whole Genome Sequence
1	SNP Array
2	Expression Array

データ入力

テストデータを入力する

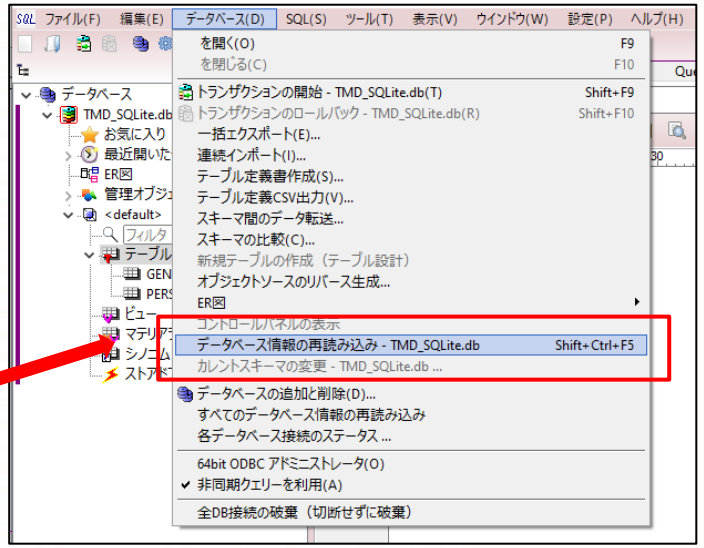
テーブル作成方法


A5:SQL Mk-2 64bit edition (Portable mode) Version 2.15.1
ファイル(F) データベース(D) SQL(S) ツール(T) 表示(V) ウインドウ(W) 設定(P) ヘルプ(H)
新規(N)... Ctrl+N
開く(O)... Ctrl+O
最近使ったファイル
ファイルの関連付け...
タブを閉じる Ctrl+W
終了(X)
ER図
管理オブジェクト
<default>
新規ドキュメント
SQL
ER図
DMonkey スクリプト
A5:SQL Mk-2 64bit edition (Portable mode) Version 2.15.1 - [Query-1]
SQL ファイル(F) 編集(E) データベース(D) SQL(S) ツール(T) 表示(V) ウインドウ(W) 設定(P) ヘルプ(H)
Query-1
データベース
TMD_SQLite.db
お気に入り
最近開いたテーブル
管理オブジェクト
<default>
TMD_SQLite.db
秒

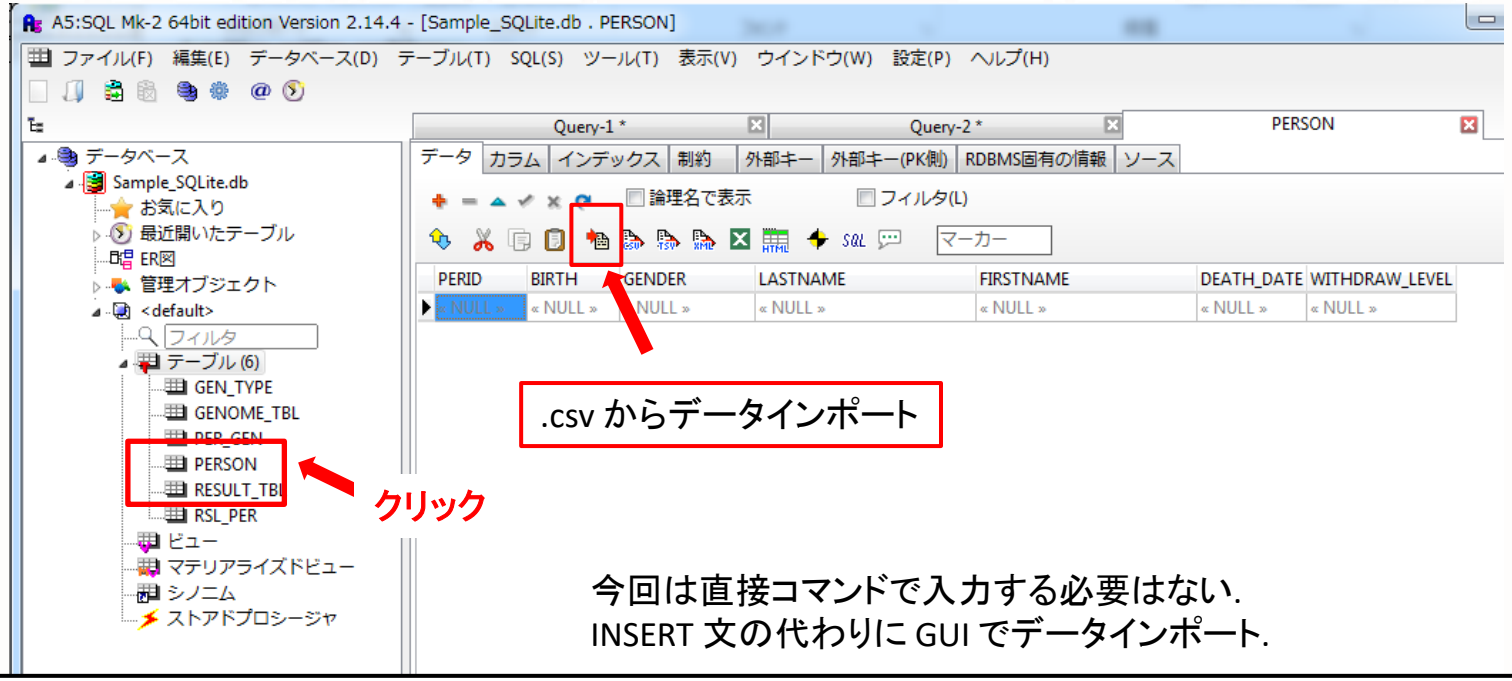


1. PERSON テーブル作成コマンドをコピーペースト

2. こちらにすぐには反映されないため、データベース読み直しを行う.



データ入力方法



今回は直接コマンドで入力する必要はない.
INSERT 文の代わりに GUI でデータインポート.

INSERT　　INTO　[テーブル名]　VALUES (..)　（WHERE　修飾）;

SQLの命令体系　4大命令

SELECT [列名]	**FROM** [テーブル名]	（**WHERE** 修飾）;
UPDATE [テーブル名]	**SET** [列名]	（**WHERE** 修飾）;
DELETE	**FROM** [テーブル名]	（**WHERE** 修飾）;
INSERT	**INTO** [テーブル名] **VALUES (..)**	（**WHERE** 修飾）;

SQL 命令に対する権限設定

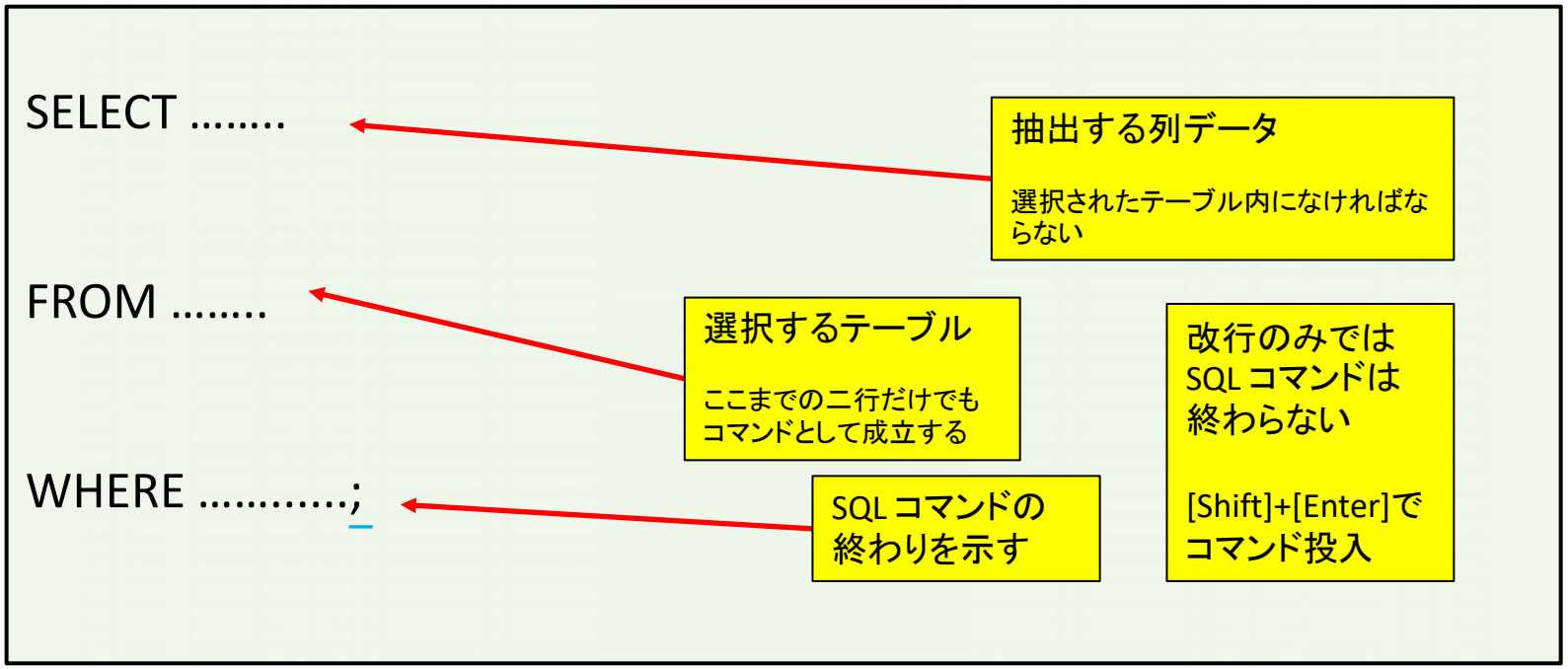
SELECT　　UPDATE　　DELETE　　INSERT

通常は、これら4つの命令に対して各ユーザーにそれぞれ異なる実行権限が設定される。管理者以外には SELECT 権限（データ参照権限）しか与えられない場合が多い。

DBMSによって異なるが、実際にはこれら以外にも多数の参照権限、実行権限に関する設定が存在し、不用意な設定変更によってデータベースが破壊されないよう設計することが可能になっている。

データ検索

SQL の命令体系　SELECT

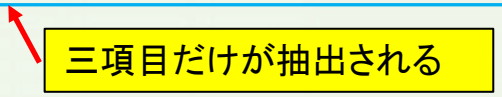


（1）

```
SELECT  *  FROM PERSON;
```

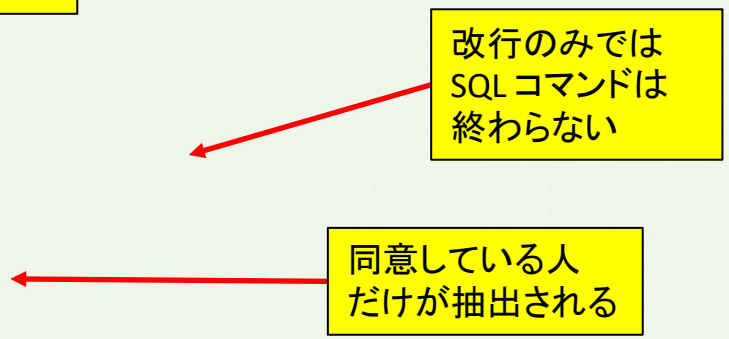
（2）

```
SELECT  PERID, LASTNAME, FIRSTNAME  FROM PERSON;
```



（3）

```
SELECT *
FROM  PERSON
WHERE  WITHDRAW_LEVEL = 0;
```



（4）

```
SELECT  *
FROM  PERSON
WHERE  BIRTH >= '1990-01-01';
```

シングルクォートで括った文字は文字列と認識される。
特定の形式で入力されている場合は（ここでは日付）その形式に対応する情報として認識される。

この日以降のデータを表示する

（5）

```
SELECT  *
FROM  PERSON
WHERE  GENDER = 2
ORDER  BY BIRTH DESC;
```

降順で表示する

昇順なら ASC を用いる

SQL の命令体系　比較演算

NULL の使い方

[列名]　IS NULL　[列名] IS NOT NULL などと表記

[列名] = NULL　　[列名] <> NULL とは書かない

（問1）
PERSON テーブルから亡くなった方を除いて抽出せよ.

（問1答）

```
SELECT  *
FROM  PERSON
WHERE  DEATH_DATE IS NULL;
```

SQL の命令体系　CASE

CASE の使い方

```
CASE  WHEN  1    THEN    '男'
      WHEN  2    THEN    '女'
      ELSE '未指定'
END
```

（問2）

PERSON テーブルの GENDER 表示を数字から漢字に変えよ.

（問2答）

```
SELECT   PERID, LASTNAME, FIRSTNAME,
         CASE GENDER
                  WHEN 1 THEN '男'
                  ELSE '女'
         END
FROM             PERSON;
```

テーブルの結合

SQL の命令体系　テーブルの結合

```
SELECT   選択リスト

FROM             テーブルA

JOIN             テーブル B
         ON      テーブルA.XXX =テーブルB.XXX
```

JOIN にはいくつかの種類があるがここでは省略.

INNER JOIN（省略するとこれになる）
OUTER JOIN
LEFT JOIN
RIGHT JOIN

ゲノムID（GEN ID）とその生産物（PRODUCT）の関係を表示

GENOME_TBL と GEN_TYPE を結合

DATATYPE の一致が結合条件

（6）

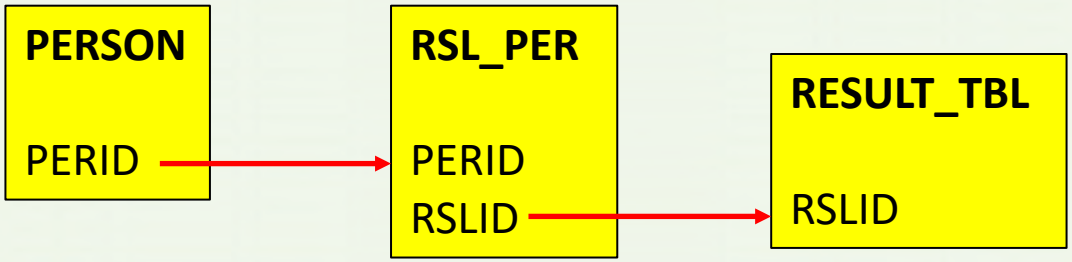
```
SELECT      GENOME_TBL.GENID, GEN_TYPE.PRODUCT,
            GENOME_TBL.PLATE_ID
FROM        GENOME_TBL
JOIN        GEN_TYPE
      ON    GENOME_TBL.DATATYPE = GEN_TYPE.DATATYPE
ORDER BY    GENOME_TBL.GENID;
```

GENOME_TBL にしかない GENID

GENOME_TBL の中にはなかった
PRODUCT が結合されている

GENID	PRODUCT	PLATE_ID
51009	Whole Genome Sequence	0926104.txt
51009	SNP Array	0194157.txt
51078	SNP Array	0802558.txt
51093	Whole Genome Sequence	0358604.txt
51093	SNP Array	0173347.txt
51093	Expression Array	0829011.txt
51158	SNP Array	0092687.txt
51197	Whole Genome Sequence	0115846.txt
51197	SNP Array	0307140.txt
51197	Expression Array	0555902.txt
51209	Whole Genome Sequence	0663537.txt
51209	Expression Array	0030794.txt
51295	SNP Array	0014007.txt
51346	SNP Array	0384000.txt
51384	Whole Genome Sequence	0451101.txt

（問7）
それぞれの参加者について、検査を行った日（検査に訪れた日）を各人日付順に抽出せよ．但し同意撤回者を除く．



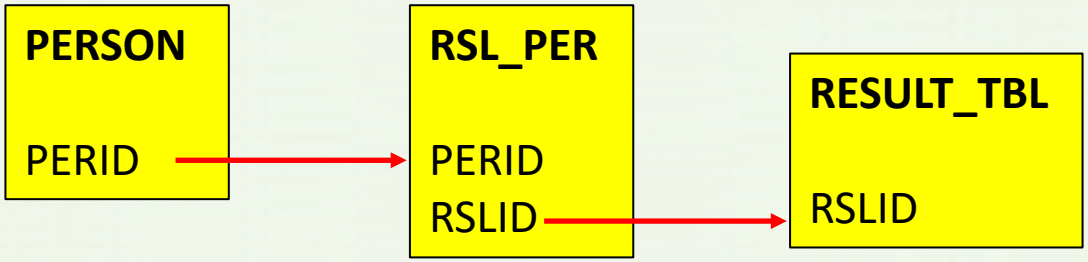
（答7）

```
SELECT   PERSON.PERID, PERSON.LASTNAME,
                  PERSON. FIRSTNAME, RESULT_TBL.RSLDATE
FROM              PERSON
JOIN              RSL_PER
         ON       PERSON.PERID = RSL_PER.PERID
JOIN              RESULT_TBL
         ON       RSL_PER.RSLID = RESULT_TBL.RSLID
WHERE             PERSON.WITHDRAW_LEVEL = 0
ORDER BY          PERSON.PERID, RESULT_TBL.RSLDATE;
```

二つのテーブルのデータが
RSL_PER テーブルを通じて結合

PERID	LASTNAME	FIRSTNAME	RSLDATE
410001	松崎	前代	2015/02/04
410001	松崎	前代	2016/05/12
410001	松崎	前代	2018/02/12
410002	加藤	真紀子	2015/02/23
410002	加藤	真紀子	2016/06/19
410002	加藤	真紀子	2018/02/28
410003	丘	清志	2015/04/10
410003	丘	清志	2017/01/03
410003	丘	清志	2018/02/28
410004	黒尾	真	2015/05/28
410004	黒尾	真	2017/01/13
410004	黒尾	真	2018/04/07

（問8）
それぞれの参加者について、検査結果が60.0以上の人を抽出せよ.
但し同意撤回者を除く.



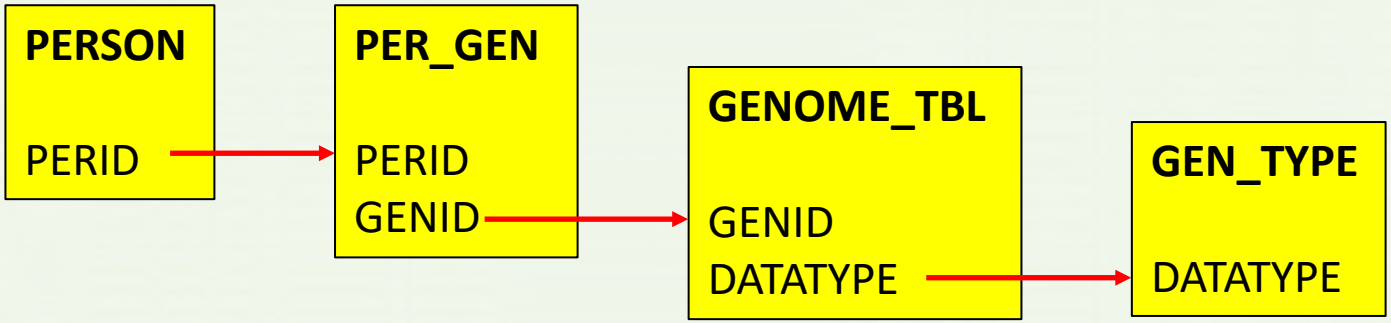
（答8）

```
SELECT   PERSON.PERID, PERSON.LASTNAME,
                  PERSON.FIRSTNAME, RESULT_TBL.RESULT
FROM              PERSON
JOIN      RSL_PER
          ON      PERSON.PERID = RSL_PER.PERID
JOIN      RESULT_TBL
          ON      RSL_PER.RSLID = RESULT_TBL.RSLID
WHERE     RESULT_TBL.RESULT >= 60.0
          AND     PERSON.WITHDRAW_LEVEL = 0
ORDER BY          PERSON.PERID, RESULT_TBL.RESULT;
```

二つのテーブルのデータが
RSL_PER テーブルを通じて結合

PERID	LASTNAME	FIRSTNAME	RESULT
410001	松崎	前代	61.89
410001	松崎	前代	67.92
410002	加藤	真紀子	85.95
410002	加藤	真紀子	88.19
410003	丘	清志	60.62
410003	丘	清志	84.84
410004	黒尾	真	68.18
410004	黒尾	真	69.78
410007	若林	邦夫	77.06
410007	若林	邦夫	83.95
410009	後藤	真樹	78.88
410009	後藤	真樹	81.81
410011	清原	竜太	64.23

（問9）
それぞれの参加者について、どのようなゲノム検査を行ったか
について抽出せよ．但し同意撤回者を除く．



（答9）

```
SELECT   PERSON.PERID, PERSON.LASTNAME,
                  PERSON. FIRSTNAME, GEN_TYPE. PRODUCT
FROM              PERSON
JOIN              PER_GEN
         ON       PERSON.PERID = PER_GEN.PERID
JOIN              GENOME_TBL
         ON       PER_GEN.GENID = GENOME_TBL.GENID
JOIN              GEN_TYPE
         ON       GENOME_TBL. DATATYPE = GEN_TYPE. DATATYPE
WHERE    PERSON.WITHDRAW_LEVEL = 0
ORDER BY          PERSON.PERID;
```

二つのテーブルのデータが
PER_GEN, GENOME_TBL
テーブルを通じて結合

PERID	LASTNAME	FIRSTNAME	PRODUCT
410001	松崎	前代	SNP Array
410002	加藤	真紀子	Whole Genome Sequence
410002	加藤	真紀子	Expression Array
410003	丘	清志	Whole Genome Sequence
410003	丘	清志	SNP Array
410003	丘	清志	Expression Array
410004	黒尾	真	Whole Genome Sequence
410004	黒尾	真	Expression Array
410005	五十里	智子	Whole Genome Sequence
410007	若林	邦夫	Whole Genome Sequence
410007	若林	邦夫	SNP Array
410009	後藤	真樹	Whole Genome Sequence
410009	後藤	真樹	SNP Array

（問10）
先ほどの問に引き続き、PRODUCT の種類ではなく何種類の方法でゲノム解析を行なったかを参加者毎にカウントせよ

（答10）

```
SELECT   PERSON.PERID, PERSON.LASTNAME,
         PERSON. FIRSTNAME, COUNT(PERSON.PERID)
FROM     PERSON
JOIN     PER_GEN
    ON   PERSON.PERID = PER_GEN.PERID
JOIN     GENOME_TBL
    ON   PER_GEN.GENID = GENOME_TBL.GENID
JOIN     GEN_TYPE
    ON   GENOME_TBL. DATATYPE = GEN_TYPE. DATATYPE
WHERE    PERSON.WITHDRAW_LEVEL = 0
GROUP BY         PERSON.PERID;
```

同じ PERID で複数のレコードが出来ているため、それをカウントした結果

PERID	LASTNAME	FIRSTNAME	COUNT(PERSON.PERID)
410001	松崎	前代	1
410002	加藤	真紀子	2
410003	丘	清志	3
410004	黒尾	真	2
410005	五十里	智子	1
410007	若林	邦夫	2
410009	後藤	真樹	3
410011	清原	竜太	1
410012	松本	百合	2
410013	佐野	博文	3
410014	近藤	和広	1
410015	加賀	知美	2
410016	米津	澄江	2
410018	大橋	博	1
410019	森	伸一	1

データサイエンス応用コース

構造化データ･トランザクション

2021/03/11

大阪大学 特任准教授
shimokawa@sigmath.es.osaka-u.ac.jp 下川和郎

講義の概要

- データベース設計
  - SQL コマンド応用
    - テーブル結合、副問い合わせ
    - データ入力インデックス作成
- 実行制御・セキュリティー
    - トランザクション、ロールバック、デッドロック
    - SQL インジェクション

インデックス作成（1）

インデックスを作成すると、DB は裏で様々な方法を使って高速に検索することができる様な仕組みを作成する. インデックスは索引と類似の仕組みで、内部に決定木などが用いられている.
（DBMS の実装系に依存する）

完全一致及び前方一致の場合にはインデックスが有効.

部分一致及び後方一致の場合には効果がない.

インデックス作成（２）

CREATE INDEX [インデックス名] ON [テーブル名（カラム名）];

DROP INDEX [インデックス名];

効果のある位置
WHERE 句の後に来る列
ORDER BY句の後に来る列
JOIN の結合条件に頻繁に登場する列

トランザクション（１）

DBMS によるトランザクション制御

トランザクションの途中で処理が中断されたり他の人の処理が割り込むことができないようにする.

途中で障害が起きれば自動的にロールバックされ、全てのトランザクションが無かったことになる.

トランザクション（２）

トランザクションの原子性

トランザクションに含まれるすべての SQL 文について、必ず
「すべての実行が完了している」
か
「一つも実行されていない」
のどちらかの状態になるよう制御する.

ex. 預金口座のお金の移動

トランザクション（ACID特性）

英語名	日本語名	概要
Atomicity	原子性	トランザクションによる一連の変更のすべてを成功か失敗のいずれかとする.
Consistency	一貫性	トランザクションの前後でデータの整合性が維持され矛盾が起こらない.
Isolation	独立性	トランザクションは相互に独立し影響を与え合わない.
Durability	対障害性	トランザクションによる変更は保存され失われることはない.

トランザクション（３）

BEGIN; （START TRANSACTION;)

INSERT INTO [table]

SELECT * FROM [table] WHERE

DELETE * FROM [table] WHERE

COMMIT;

BEGIN と COMMIT の間の中断はロールバックされる.

トランザクション（４）　同時実行時の問題

あるユーザーの書き込み途中で他のユーザーが読み出す場合に起こる副作用.

潜在的問題	概要
ダーティーリード	コミットされていない未確定データを読み込む可能性. その後キャンセルされるかも知れない未確定の情報を元に別の処理を行ってしまう.
反復不能読み取り	データ読み取り後に他の人がデータを書き換え、再読み取り時に異なる結果が返ってくる.
ファントムリード	データ読み取り後に他の人にデータを追記されることによる副作用.

トランザクション（4’）ロック

トランザクション実行の際、内部でロックをかける事によって
他のユーザーからデータの読み書きができないようにする.

SET TRANSACTION ISOLATION LEVEL [分離レベル]

([分離レベル] =
[READ UNCOMMITTED]
[READ COMMITTED]
[REPEATABLE READ]
[SERIALIZABLE])

（DBMS 依存機能）

トランザクション（4’）ロック

分離レベル	ダーティーリード	反復不能読み取り	ファントムリード
READ UNCOMMITTED	○（発生する）	○	○
READ COMMITTED	×（発生しない）	○	○
REPEATABLE READ	×	×	○
SERIALIZABLE	×	×	×

トランザクション（4’）並列実行制御

一部 DBMS では、READ UNCOMMITTED が存在しない.
それら DBMS では、コミットされていない情報は読み出せないように作られている.

→ あるトランザクションによってデータが書き換えられている最中も書き換え前の情報が残っていて、他から読み出し要求が起きた場合には書き換え前の情報が提供されるようになっているため. この二つのバージョンを並存させる仕組みを並列実行制御 MVCC（Multi Version Concurrency Control）という.

（DBMS 依存機能）

トランザクション（4’）デッドロック

デッドロックの発生

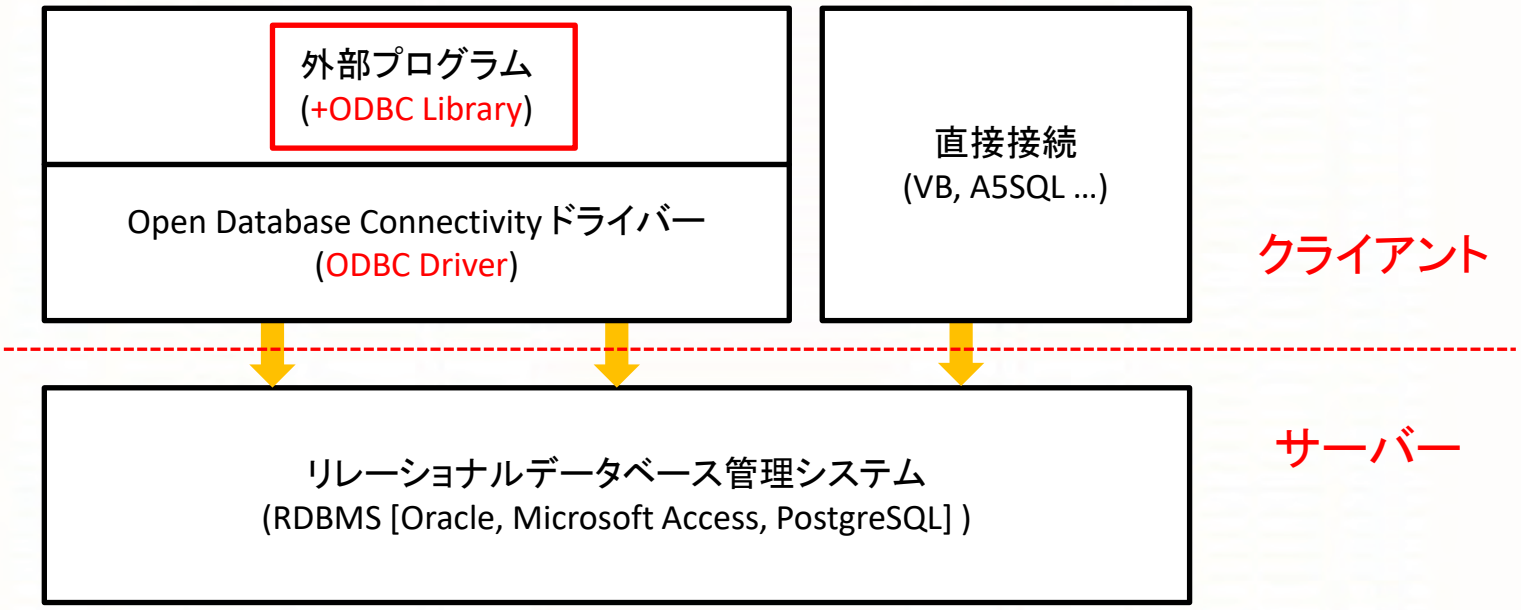
「X」をロックしたトランザクション A が、次に「Y」もロックしようとしている一方で、「Y」をロックした別のトランザクション B が、次に「X」もロックしようとするとき、「デッドロック」が発生する.

DBMS は定期的にデッドロックを調査し、発見した場合は片方のトランザクションを強制的に失敗させる事によってデッドロックを解消する。

（DBMS 依存機能）

付録：外部プログラムからのDBアクセス

外部プログラムからDBにアクセスする



各RDBMS 用 ODBC Driver

- Oracle
  - https://www.oracle.com/jp/database/technologies/instant-client/downloads.html
- PostgreSQL
  - https://www.postgresql.org/ftp/odbc/versions/msi/
- MySQL
  - https://www.mysql.com/jp/products/connector/
- Maria DB
  - https://downloads.mariadb.org/connector-odbc/

外部プログラムからの呼び出し (Perl)

https://qiita.com/arachan@github/items/34cedba6a60accf87b10

```
use DBI;

my $dbh=DBI->('dbi:ODBC:$DSN',' user','password') or $!;
my $sth=$dbh->prepare("select * from PERSON where ID='1000125'") or $dbh-
>errstr;
```

データサイエンス応用コース

分散型データベース

2021/03/11

大阪大学 特任准教授
shimokawa@sigmath.es.osaka-u.ac.jp 下川和郎

講義の概要

- トランザクション処理の現場
  - 手作業から電子的決済システムへ（1973～）
  - 中央集権型DB
  - 銀行における決済処理
- 分散型DB：ブロックチェーン技術
  - ハッシュ関数、公開鍵暗号化法
  - P2P ネットワーク

トランザクション

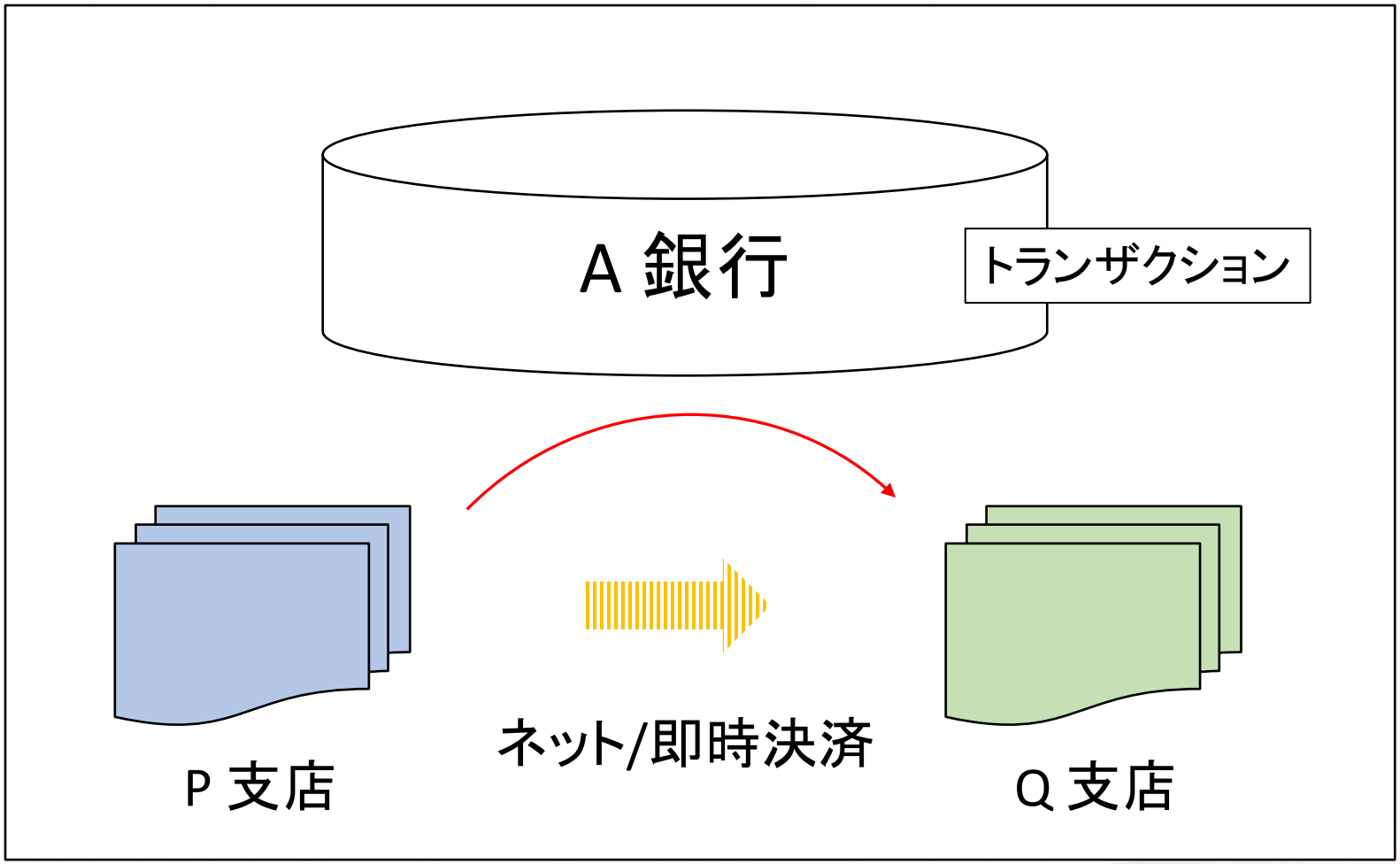
DBMS によるトランザクション制御

途中で処理が中断されたり他の処理が割り込むことができない一連の処理.

実行途中で障害が起きれば自動的にロールバックされ、全てのトランザクションが無かったことになる.

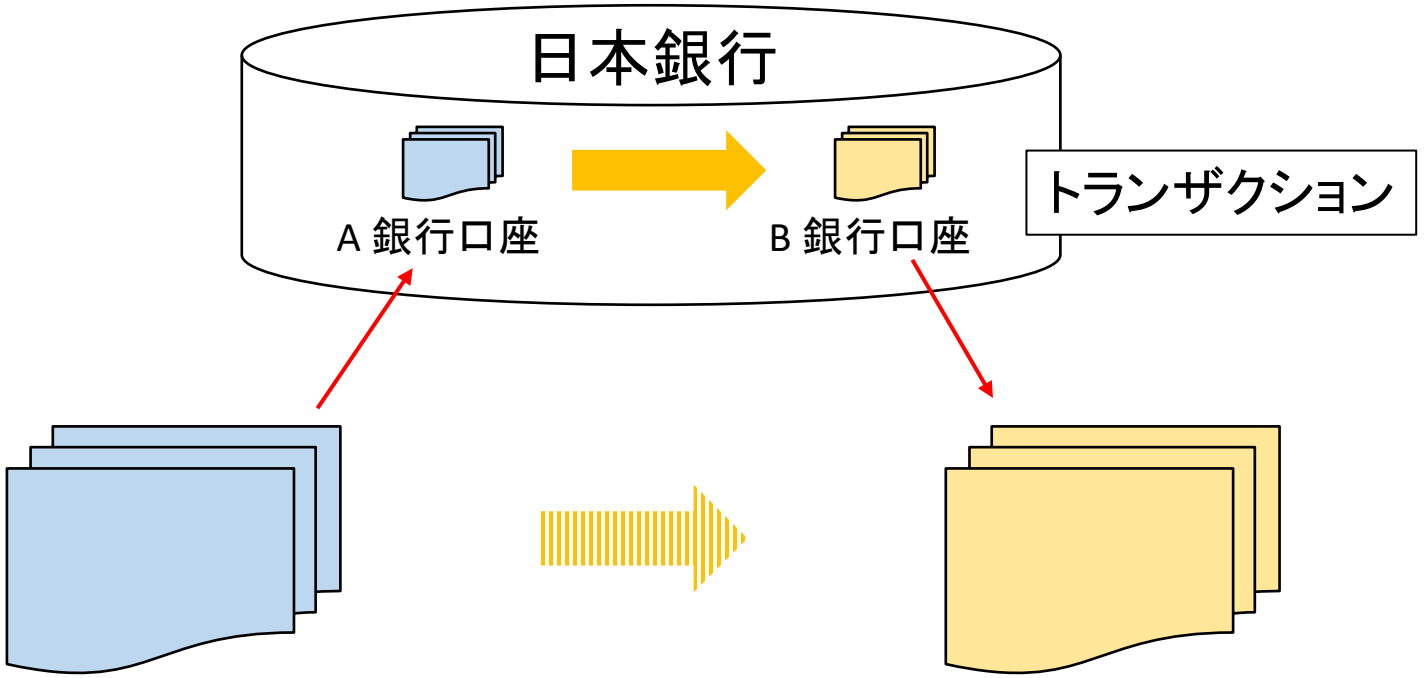
ex. 銀行間送金処理等

決済（同一銀行内）



ネット決済：一日のうちで起こった決済を合計して差額を支店間で決済する

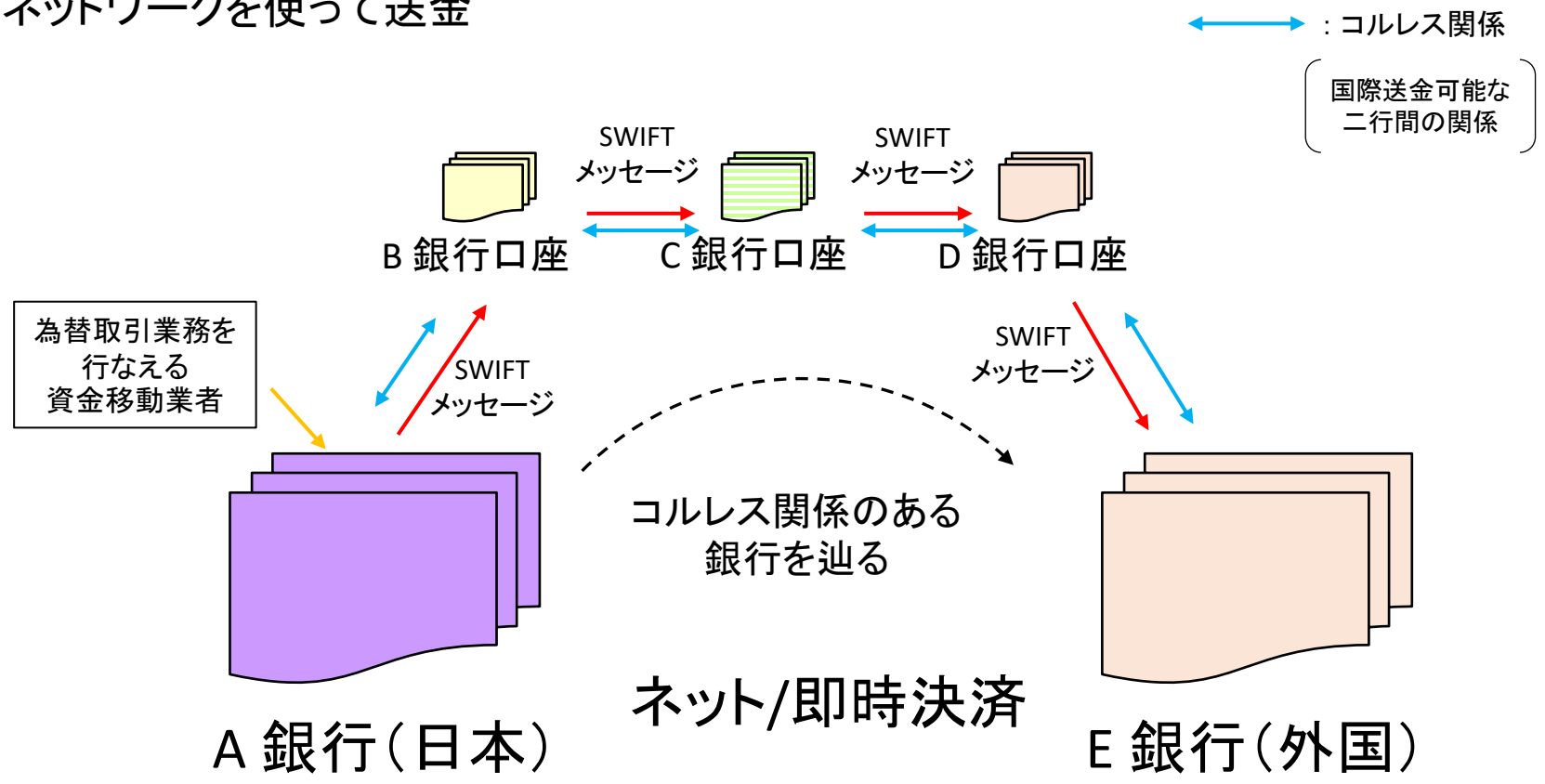
決済　日本国内



RTGS（Real-time Gross Settlement）システム

決済　国際送金（外為決済システム）

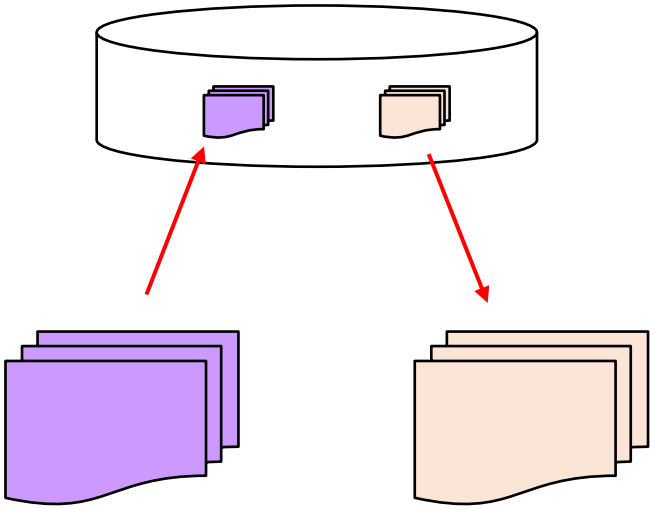
SWIFT（国際銀行間通信協会）
ネットワークを使って送金



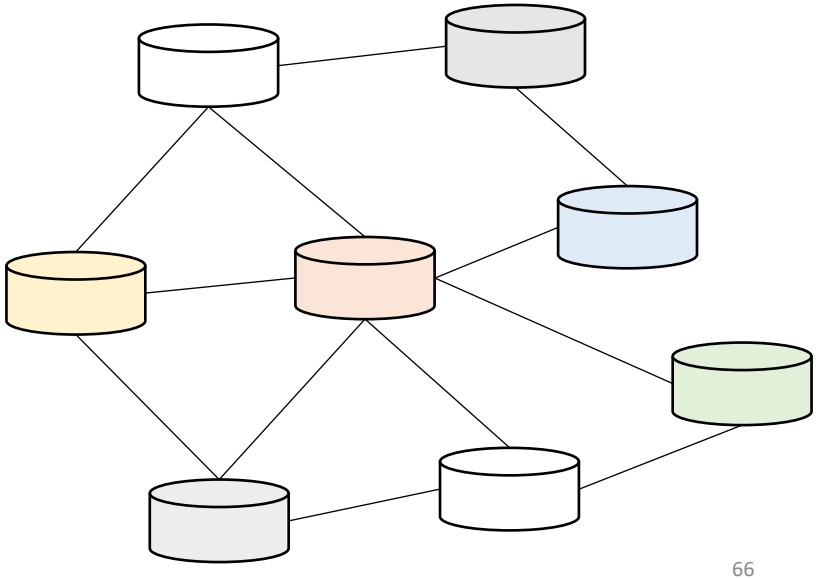
SWIFT：銀行間の国際的決済ネットワーク（ベルギー）

中央銀行決済 vs 分散システム決済

- SWIFT を使った国際決済
  - 国際的な信用、長い実績
  - 中央集権型データベース
  - 管理者権限であらゆる操作が可能



- P2P 分散ネットワーク
  - P2P データストレージを用いた分散型データベース
  - 安全で信頼性が高い
  - 秘密裏にデータを書き換えられない



ブロックチェーン（分散型台帳）

- 基本となる技術
  - ハッシュ関数
    - 一方向性関数
    - ファイルの破損、改ざんの検知
    - シノニム
  - 公開鍵暗号化法
    - 暗号業界における重大発明
    - 秘密鍵と公開鍵のペアで使われる
    - デジタル署名

ハッシュ関数

- ハッシュ関数
  - あるデータをハッシュ値（固定長）に変換する。
  - 誰が行なっても同じデータからは同じハッシュ値ができる。
  - 少しでも異なるデータからは異なるハッシュ値が生成される。
  - ハッシュ値から元データを推測できない。
  - 巨大データからも計算できる。

  - 攻撃法
    - レインボー攻撃
      - あらかじめよく使いそうなフレーズを変換した辞書を作成する

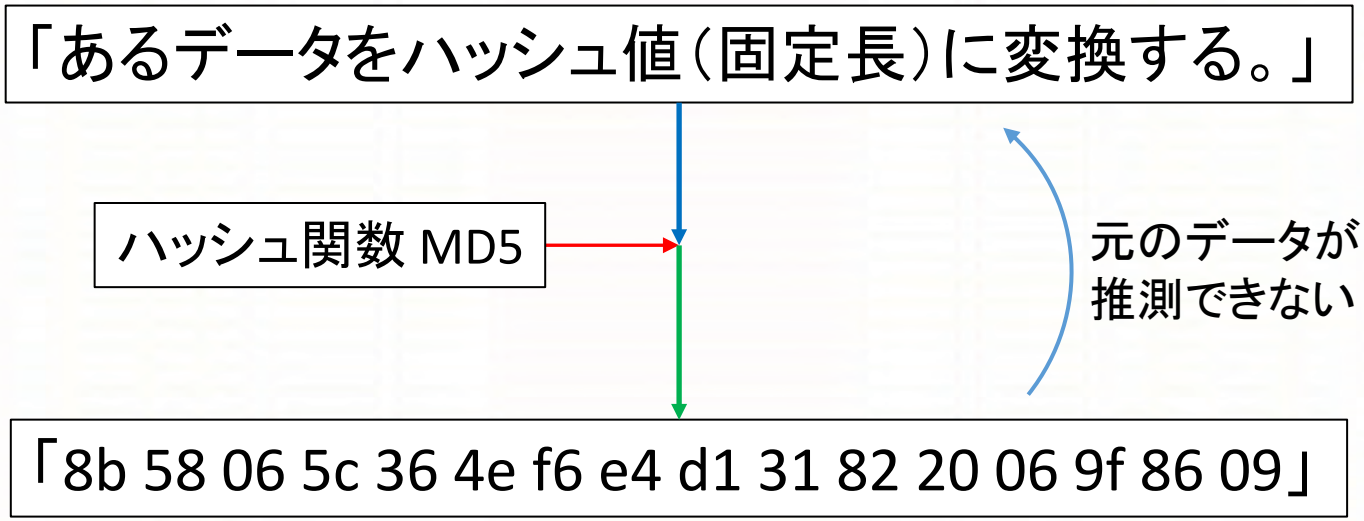
ハッシュ関数

・実行例

>CertUtil -hashfile test.txt MD5

```
C:¥Users¥K.Shimokawa>type test.txt
あるデータをハッシュ値（固定長）に変換する。

C:¥Users¥K.Shimokawa>CertUtil -hashfile test.txt MD5
MD5 ハッシュ（ファイル test.txt):
e8 3a 44 a1 f7 ea cf a7 a7 bd a7 d0 60 d7 61 38
CertUtil: -hashfile コマンドは正常に完了しました。

C:¥Users¥K.Shimokawa>type test2.txt
あるデータをハッシュ値（固定長）に変換するよ。

C:¥Users¥K.Shimokawa>CertUtil -hashfile test2.txt MD5
MD5 ハッシュ（ファイル test2.txt):
8b 58 06 5c 36 4e f6 e4 d1 31 82 20 06 9f 86 09
CertUtil: -hashfile コマンドは正常に完了しました。
```

ハッシュ関数



ハッシュ値は固定長だが、元のデータの長さは問わない

→ 異なるデータから同じハッシュ値ができる場合がある（衝突）

→ 耐衝突性の高いアルゴリズムの研究

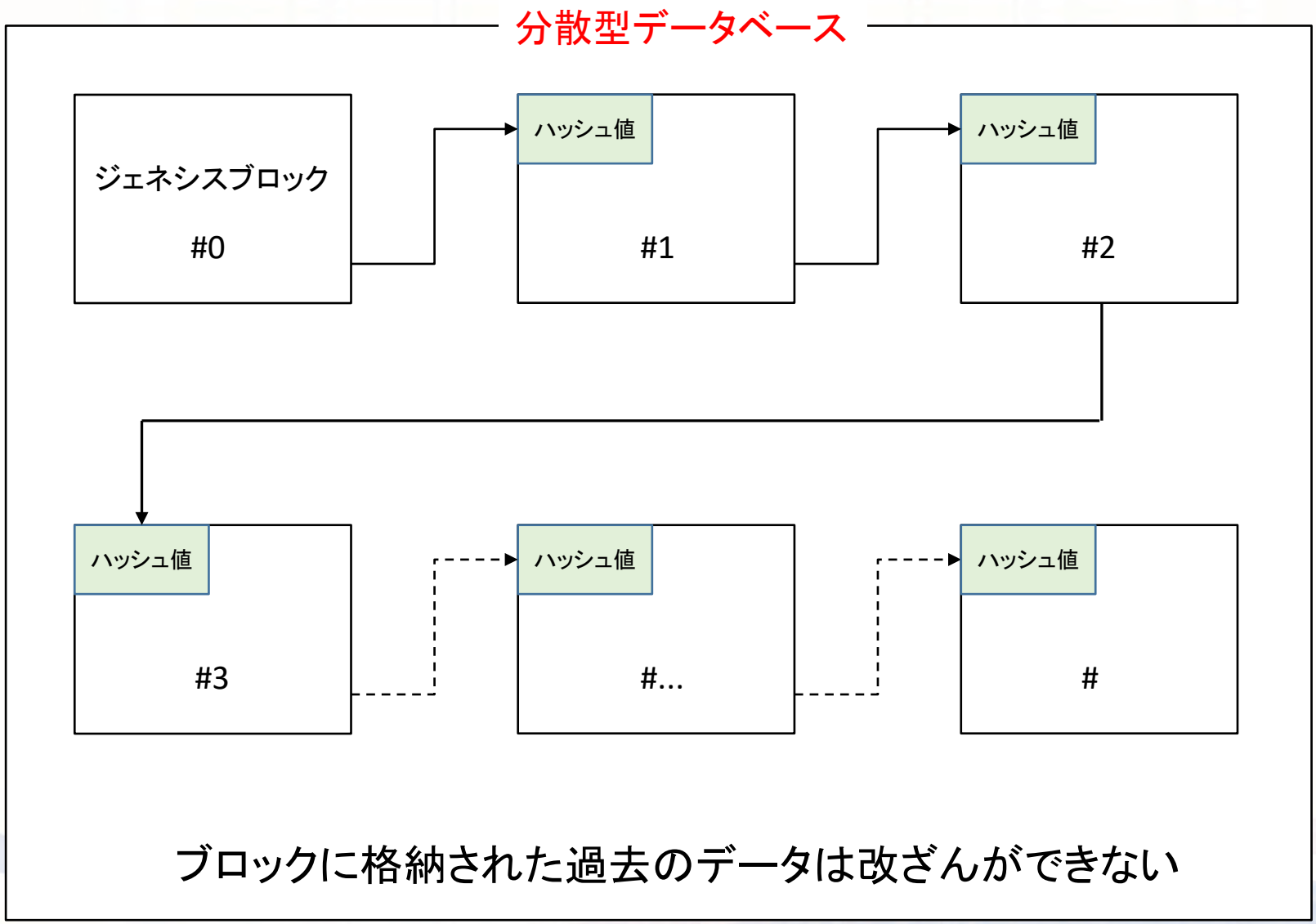
```
C:¥Users¥K.Shimokawa>CertUtil -hashfile test.txt MD4
MD4 ハッシュ (ファイル test.txt):
2c ee d9 a9 1a 93 95 bc d4 b4 98 4c c3 75 ed 4d
CertUtil: -hashfile コマンドは正常に完了しました。

C:¥Users¥K.Shimokawa>CertUtil -hashfile test2.txt MD5
MD5 ハッシュ (ファイル test2.txt):
8b 58 06 5c 36 4e f6 e4 d1 31 82 20 06 9f 86 09
CertUtil: -hashfile コマンドは正常に完了しました。

C:¥Users¥K.Shimokawa>CertUtil -hashfile test.txt SHA1
SHA1 ハッシュ (ファイル test.txt):
d3 1f f1 4b 7e 5a 21 23 cb 55 ff 24 2b b7 13 14 00 80 12 73
CertUtil: -hashfile コマンドは正常に完了しました。

C:¥Users¥K.Shimokawa>CertUtil -hashfile test.txt SHA256
SHA256 ハッシュ (ファイル test.txt):
67 34 f7 20 c8 dc b2 ef 66 dc 09 01 cc be e9 aa 92 2f f4 04 4a af 99 fb 80 67 51
 89 eb 26 87 3a
CertUtil: -hashfile コマンドは正常に完了しました。

C:¥Users¥K.Shimokawa>CertUtil -hashfile test.txt SHA512
SHA512 ハッシュ (ファイル test.txt):
e1 8a e6 0d 6e 64 44 df d1 97 3b f1 7b 5c 0c de 42 f1 53 69 6b d3 01 c3 0a cc 01
 3c d9 fa 3d c6 14 66 fd de d3 49 4f 3a 96 a5 1d 06 cf d1 1c 3e 29 88 db 1e e4 8
f d1 e7 d4 63 59 3d 85 8d 5e 53
CertUtil: -hashfile コマンドは正常に完了しました。
```

ハッシュ関数

・ハッシュ関数の種類と結果

開発年	関数名	変換例
1990年	MD4	2c ee d9 a9 1a 93 95 bc d4 b4 98 4c c3 75 ed 4d
1992年	MD5	8b 58 06 5c 36 4e f6 e4 d1 31 82 20 06 9f 86 09
1995年	SHA1	d3 1f f1 4b 7e 5a 21 23 cb 55 ff 24 2b b7 13 14 00 80 12 73
2001年	SHA2-256	67 34 f7 20 c8 dc b2 ef 66 dc 09 01 cc be e9 aa 92 2f f4 04 4a af 99 fb 80 67 51 89 eb 26 87 3a
2001年	SHA2-512	e1 8a e6 0d 6e 64 44 df d1 97 3b f1 7b 5c 0c de 42 f1 53 69 6b d3 01 c3 0a cc 01 3c d9 fa 3d c6 14 66 fd de d3 49 4f 3a 96 a5 1d 06 cf d1 1c 3e 29 88 db 1e e4 8f d1 e7 d4 63 59 3d 85 8d 5e 53

ブロックチェーン

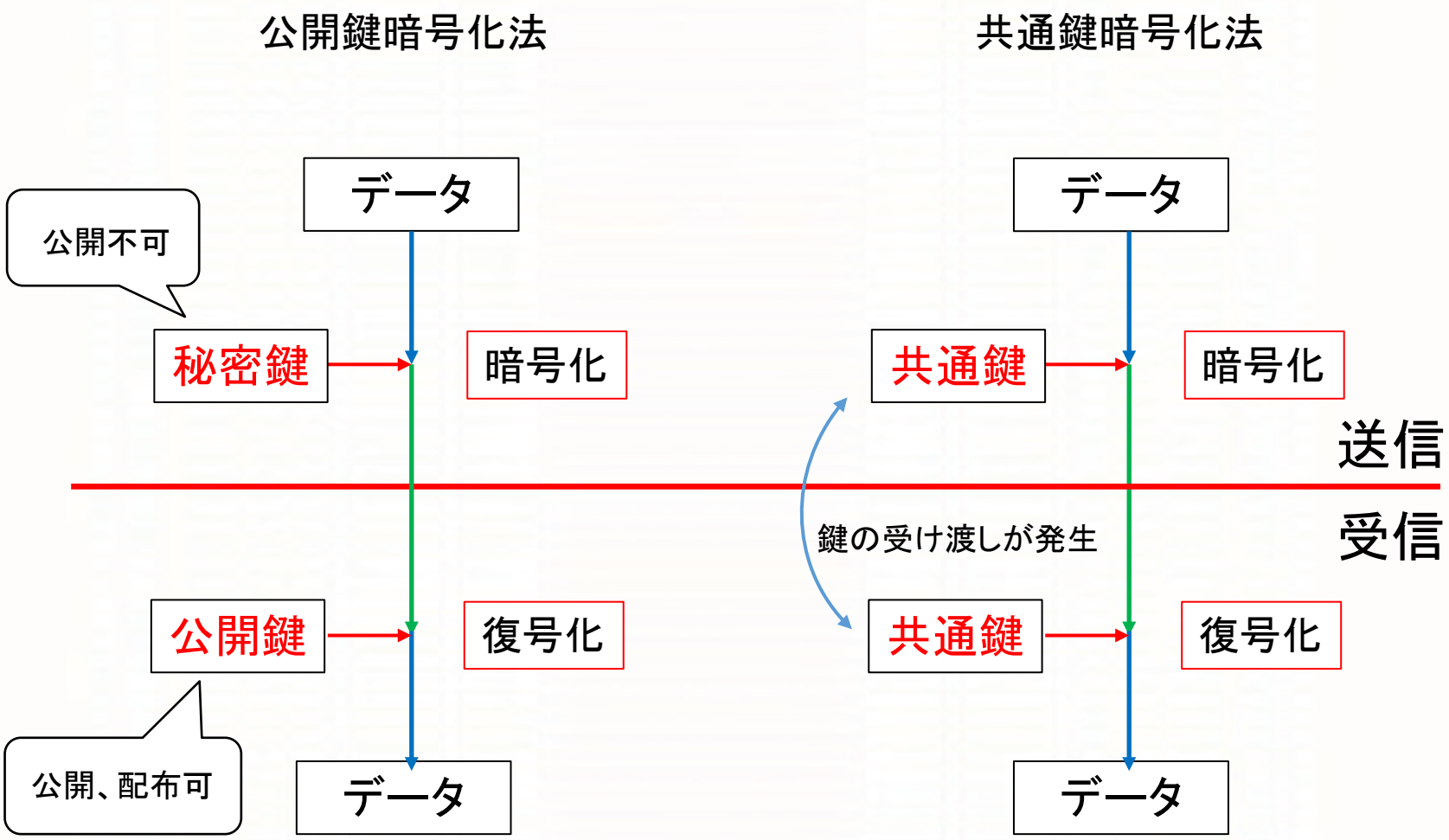

分散型データベース
ジェネシスブロック
#0
ハッシュ値
#1
ハッシュ値
#2
ハッシュ値
#3
ハッシュ値
#...
ハッシュ値
#
ブロックに格納された過去のデータは改ざんができない

公開鍵暗号化法

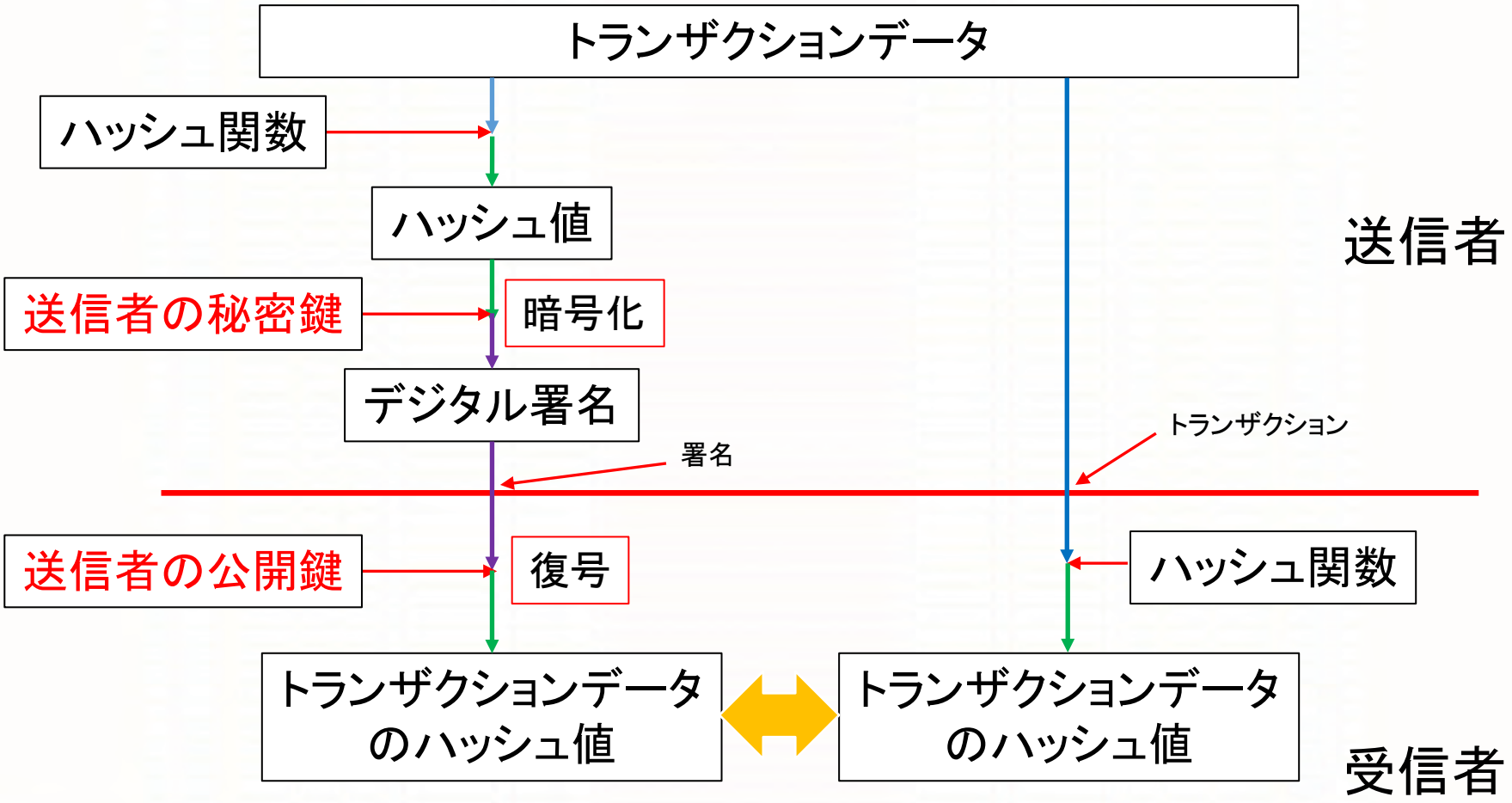
- 共通鍵暗号化法
  - 暗号化と複合化の鍵が共通
    - 鍵を入手すると誰でも復号化できる
  - 暗号化法の種類
    - DES
    - AES

- 公開鍵暗号化法
  - 「秘密鍵」と「公開鍵」の二つがペアになる
    - 秘密鍵で暗号化して公開鍵で復号する
    - 公開鍵は公開しても構わない
    - 巨大データの暗号化に向かない（計算量）
  - 暗号化法の種類
    - RSA（素因数分解暗号）
    - ECDSA（楕円曲線暗号）

動画などのファイルの実体は送れない


公開鍵暗号化法
共通鍵暗号化法
データ
公開不可
秘密鍵
暗号化
データ
共通鍵
暗号化
送信
受信
鍵の受け渡しが発生
公開鍵
復号化
公開、配布可
データ
共通鍵
復号化
データ
公開鍵で復号可能
→秘密鍵を持った人が
暗号化した
共通鍵が奪われると、どこから
来たのか証明ができなくなる。

デジタル署名の原理を利用 （ブロックチェーン）



これらを比較して一致すれば、送信者が
トランザクションデータを作成したことが証明できる

新しいブロックを作成するときハッシュ値の整合性を検証する
過去のブロックは改ざんができない

誰が何処に送金

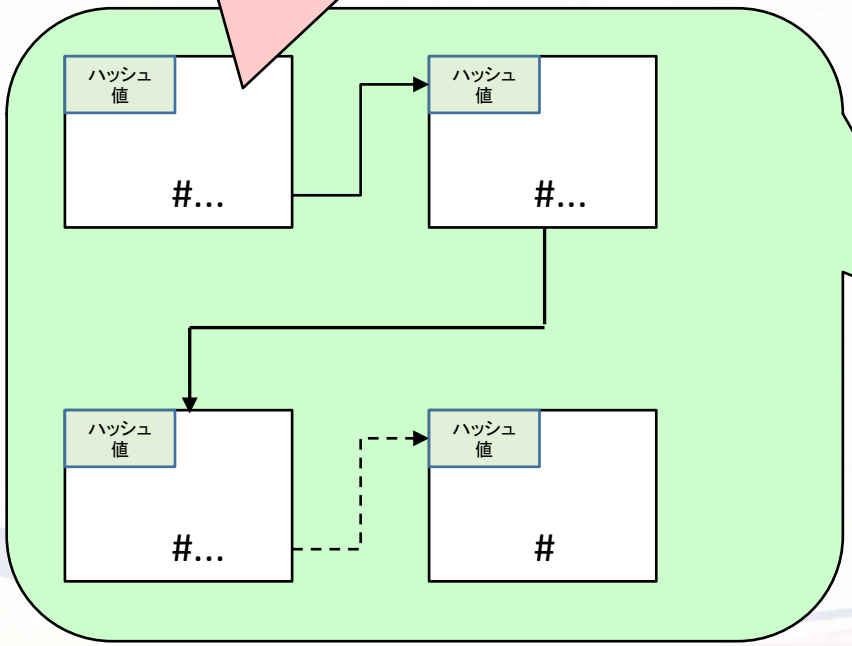
トランザクション
デジタル署名

トランザクション
デジタル署名

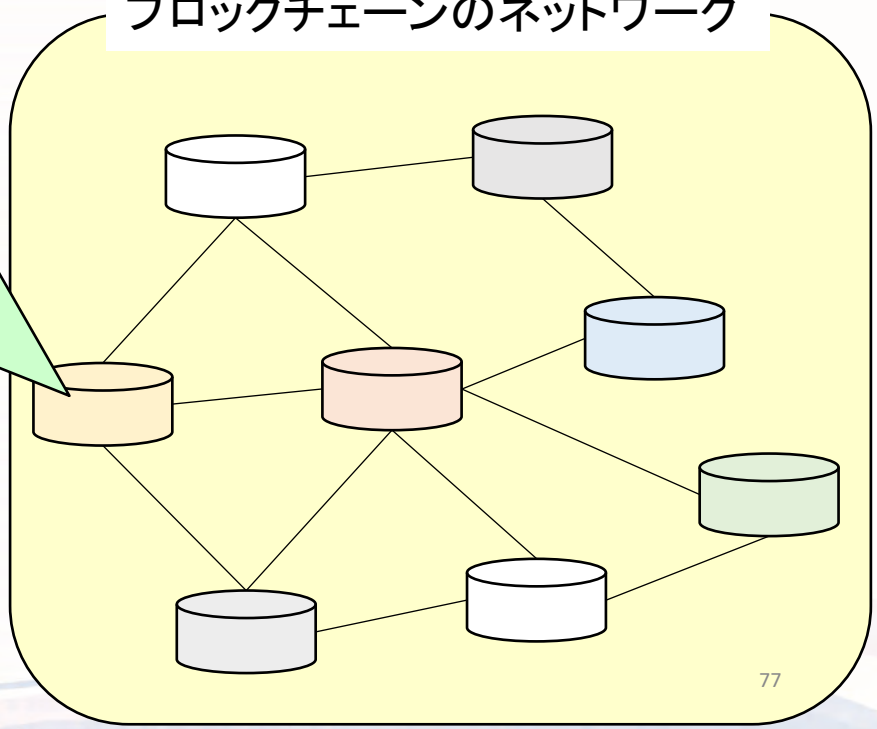
トランザクション
デジタル署名

トランザクション
デジタル署名

画像や大きなデータを送りたい場合は、
それらの実体を他の P2P ネットワーク
へ保存することも可能.
名前は実体のハッシュ値を用いる.



ブロックチェーンのネットワーク



ブロックチェーン

- ユーザーは取引所を通じてブロックチェーンにトランザクションを送り込む
- トランザクションを集めてブロックに入れる作業はマイナーが行う
- 通貨、証券、証拠、権利書、ポイント、クーポン 等
- やり取りの際のウォレット（財布）は自分の公開鍵そのもの
- 巨大なデータについては名前を付けて他の P2P NW へ

現状

- 現状では（ブロックチェーンの応用としての）仮想通貨等は決済に向いていない（価値の変動が激しい）
- 見られては困る情報、巨大データについてはブロックチェーンの中に含められない
- 改ざんのできない公開情報の蓄積としては価値が高いため、様々な応用が考えられる

応用例

- 学歴職歴データベース
- 官公庁の公証プラットフォーム
- ゲーム内アイテム、キャラクターの売買
- 商品流通情報の可視化
- …

研究発表

本コース受講によるスキル伸長を自己で確認する機会として、また、業務に対する自己のアイデアを精錬する機会として、研究発表の場を活用する。

研究発表テーマ「データサイエンス・AIを用いた新規サービスの提案」

- 内容は、本コースで学習した技術を活用するものとすること
- 各回演習での検討・発表を再編成・統合するものでも可
- 各回演習とは別に、新たなトピックを選択し、検討するものでも可