# データベース
# 第7回

土田　隼之

授業計画

| | 週 | 授業内容・方法 | 週ごとの到達目標 |
|---|---|---|---|
| 後期 | 1週 | データベースの概要 | データベースの役割、データベースの学術利用、業務利用、その意義と用途を理解できる。 |
| | 2週 | データベースのための基礎理論 | 集合とその演算、組（タプル）、組の集合としてのリレーションなど、データベースのための基礎理論を理解できる |
| | 3週 | リレーショナルデータモデルとリレーショナル代数 | RDBMSで利用されるデータモデルであるリレーショナルデータモデルとデータ操作のためのリレーショナル代数を理解できる。 |
| | 4週 | SQL(1) | RDBMSの利用全般に用いられる言語SQLの基本を理解できる。リレーションへのデータ登録・削除・更新、簡単な問合せなど、基本的なSQLの使い方を理解できる。 |
| | 5週 | SQL(2) | RDBMSの利用全般に用いられる言語SQLを作成できる。SQLにおける問合せを行うselect文を理解できる。 |
| | 6週 | RDBMSの内部構成 | RDBMSの内部構成、および大量のデータの中から目的とするデータに素早くアクセスする仕組みであるインデックスを理解できる。 |
| | 7週 | 問合せ最適化 | RDBMSで、SQL問合せを実行するための実行プランを生成するための問合せ最適化が理解できる。 |
| | 8週 | 中間試験 | 中間試験 |
| | 9週 | プログラムからの | |
| | 10週 | 正規化 | およる。 |
| | 11週 | データモデリング | ータ項目を抽出・整理して適切なデータ構造を決定する作業であるデータモデリングが理解できる。 |
| | 12週 | SQL(3) | RDBMSの利用全般に用いられる言語SQLを作成できる。SQLにおける問合せを行う高度なselect文を理解できる。 |
| | 13週 | トランザクションと同時実行制御 | アプリケーションがデータベースにアクセスする単位であるトランザクションの概念、および複数のトランザクションを正常に実行するための基礎理論を理解できる。 |
| | 14週 | NoSQLデータベースとビッグデータ(1) | ビッグデータを扱うため開発された新しいデータベースであるNoSQLの基礎を理解できる。主にNoSQLの概観と、ビッグデータを扱うためのデータモデルや実行制御理論を理解できる。 |

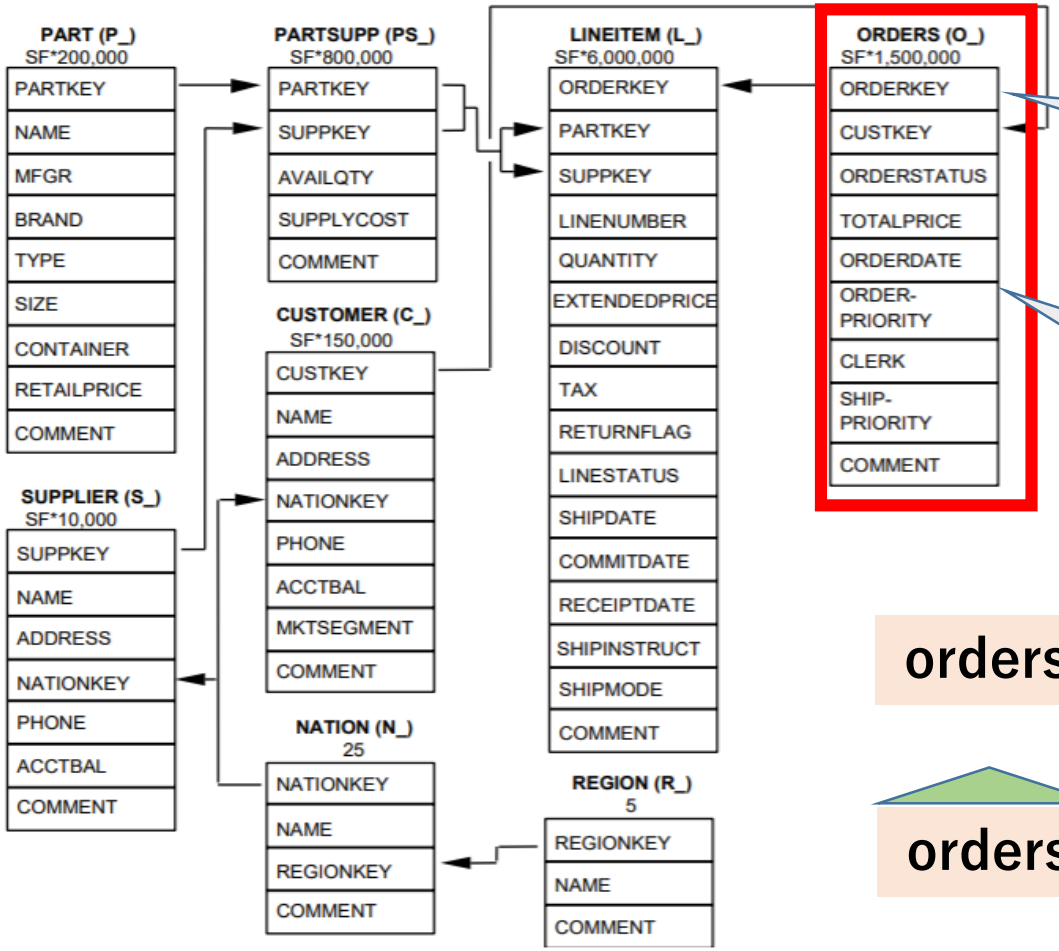
**模擬試験:ポートフォリオ点に加算。課題だけで、ポートフォリオ点は満点になる。**
**#模擬試験は、退出前に土田に提出ください**

# 課題6-1
# 締切:11/23

下記データに対して、SF=1 の時に a)～c)の SQL に対する最短処理時間を記載せよ。小数点一桁まで記載すること。各表のレコード数は表名下記の数式(例えば、LINEITEM なら SF*6,000,000[行])で定まるとし、各テーブルは、カラム数やカラム型に関係なく 1,000,000[行]で 1[GB]とする。シーケンシャル READ は 0.5[GB/s]とし、索引アクセスには 1 レコード毎に 0.0001[s]かかるとする。なお、読込みディスク I/O 以外の処理(例えば、CPU 処理)は無視できるものとする。(4 問×5 点, SQL 問題:10 点)

a) select * from orders;で表スキャンを前提とした処理時間

b) select * from orders;で索引アクセスを前提とした処理時間(全件を 1 レコード毎に索引アクセス)と算出式を記載せよ。




ORDERS表はSF1で1,500,000行
→1,500,000[行]/1,000,000[行]=1.5[GB]

問a)1.5[GB]を0.5[GB/s]で表スキャンすると何秒か？
問b)1,500,000[行]を1レコード毎に0.0001[s]かけて索引アクセスすると何秒か？

orders

orders

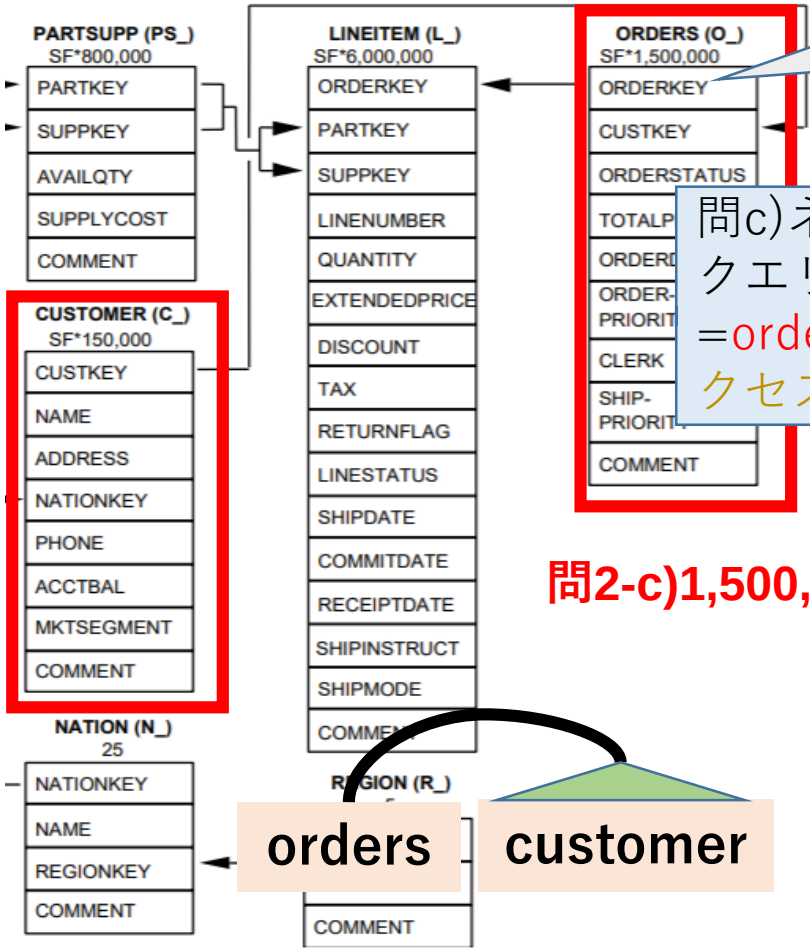
問a)1500000[行]/1000000[GB/行]/0.5[GB/秒]=3秒
問b)1500000[行]*0.0001[秒/行]=150秒

# 課題6-2
# 締切:11/23

下記データに対して、SF=1 の時に a)～c)の SQL に対する最短処理時間を記載せよ。小数点一桁まで記載すること。各表のレコード数は表名下記の数式(例えば、LINEITEM なら SF*6,000,000[行])で定まるとし、各テーブルは、カラム数やカラム型に関係なく 1,000,000[行]で 1[GB]とする。シーケンシャル READ は 0.5[GB/s]とし、索引アクセスには 1 レコード毎に 0.0001[s]かかるとする。なお、読込みディスク I/O 以外の処理(例えば、CPU 処理)は無視できるものとする。(4 問×5 点, SQL 問題:10 点)

orders 表と customer 表にそれぞれ custkey 索引が備わっており、orders 表では 1 つの custkey に対して 10 レコード存在するときに、以下 SQL の処理時間と算出式を記載せよ。

c) select * from orders, customer where orders.custkey=customer.custkey 外表(駆動表)が orders 表




ORDERS表はSF1で1,500,000行
→1,500,000[行]/1,000,000[行]=1.5[GB]

問c)ネストループ結合で、外部表がorders表なので
クエリ処理時間=外部表読込み時間+(外部表レコード×絞込み)×内部表読込み時間
=orders表の最短表読込み時間+(orders表のレコード数×1)×customer表の最短アクセス時間

**O表は、a)b)で表スキャンが早いとわかっている**

**問2-c)1,500,000[行]/1,000,000[行/GB]/0.5/[GB/秒]+1,500,000[行]*0.0001[秒/行]=153秒**

**C表は、索引アクセスのほうが高速**

**C表 1回の表スキャン:150,000[行]/1,000,000[行/GB]/0.5[GB/秒]=0.3[秒]**
**O表行数回のC表スキャン=1,500,000[行]*0.3[秒]=45000[秒]**
**O表行数回のC表索引アクセス=1,500,000[行]*0.0001[秒]=150[秒]**

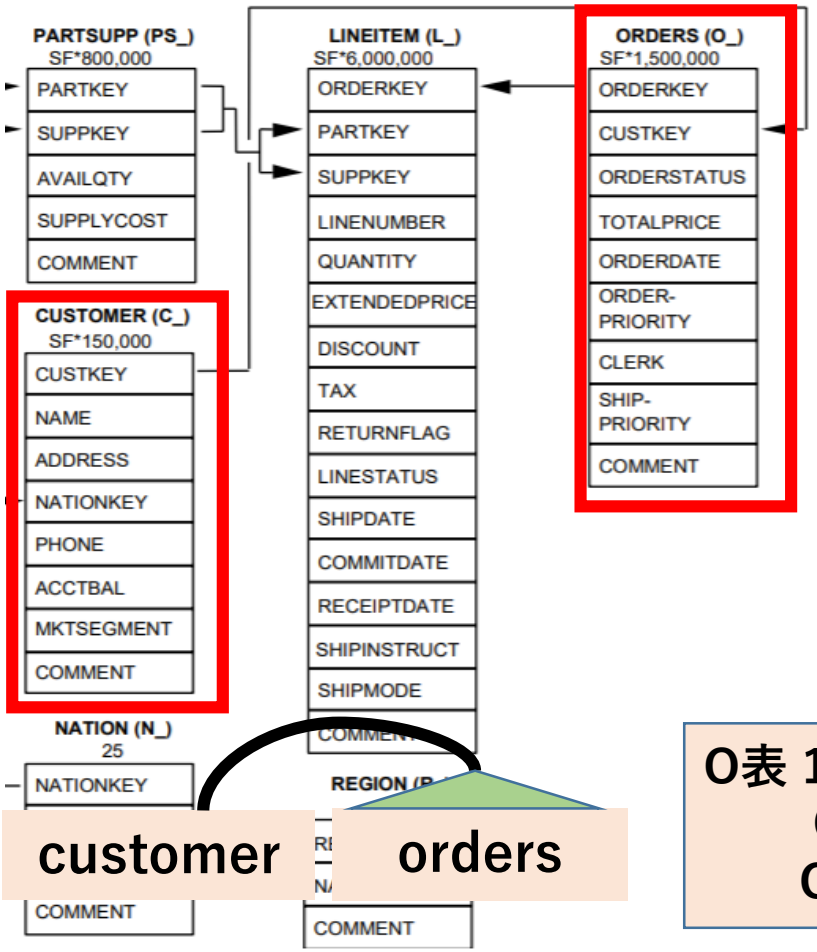
**orders** **customer**

# 課題6-3
# (余裕あれば)
# 締切:11/23

下記データに対して、SF=1 の時に a)～c)の SQL に対する最短処理時間を記載せよ。小数点一桁まで記載すること。各表のレコード数は表名下記の数式(例えば、LINEITEM なら SF*6,000,000[行])で定まるとし、各テーブルは、カラム数やカラム型に関係なく 1,000,000[行]で 1[GB]とする。シーケンシャル READ は 0.5[GB/s]とし、索引アクセスには 1 レコード毎に 0.0001[s]かかるとする。なお、読込みディスク I/O 以外の処理(例えば、CPU 処理)は無視できるものとする。(4 問×5 点, SQL 問題:10 点)

orders 表と customer 表にそれぞれ custkey 索引が備わっており、orders 表では 1 つの custkey に対して 10 レコード存在するときに、以下 SQL の処理時間と算出式を記載せよ。

d)select * from orders,customer where orders.custkey=customer.custkey 外表(駆動表)が customer 表




customer　orders

問d)ネストループ結合で、外部表がcustomer表なので
クエリ処理時間=外部表読込み時間+(外部表レコード×絞込み)×内部表読込み時間
=customer表の最短表読込み時間+(customer表のレコード数×1)×orders表の最短アクセス時間

**C表 1回の表スキャン:150,000[行]/1,000,000[行/GB]/0.5[GB/秒]=0.3[秒]**
**C表行数回のC表索引アクセス=150,000[行]*0.0001[秒]=15[秒]**
**→表スキャンが早い**

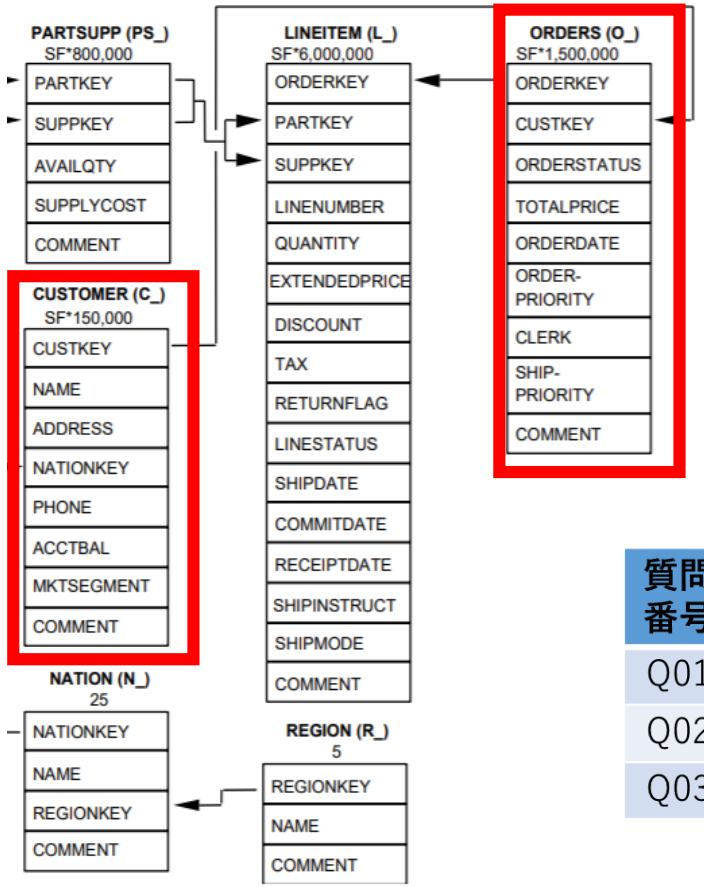
**150000[行]/1000000[行/GB]/0.5[GB/秒]**
**+150000[行]*10[ファンアウト]*0.0001[秒/行]=150.3秒**

**O表 1回の表スキャン:1,50,000[行]/1,000,000[行/GB]/0.5[GB/秒]=3[秒]**
**C表行数回のO表スキャン=150,000[行]*3[秒]=450,000[秒]**
**C表行数回のO表索引アクセス=150,000[行]*0.0001[秒]*10=150[秒]**

# 課題6-3
# (余裕あれば)
# 締切:11/23

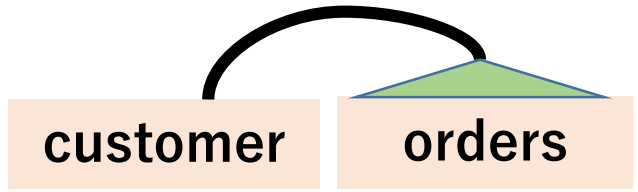
下記データに対して、SF=1 の時に a)～c)の SQL に対する最短処理時間を記載せよ。小数点一桁まで記載すること。各表のレコード数は表名下記の数式(例えば、LINEITEM なら SF*6,000,000[行])で定まるとし、各テーブルは、カラム数やカラム型に関係なく 1,000,000[行]で 1[GB]とする。シーケンシャル READ は 0.5[GB/s]とし、索引アクセスには 1 レコード毎に 0.0001[s]かかるとする。なお、読込みディスク I/O 以外の処理(例えば、CPU 処理)は無視できるものとする。(a)～c)問×5 点, SQL 問題:10 点)

索引アクセスには(アクセス先に格納されている)1レコード毎に0.0001[s]かかる
→0.0001[s/アクセスレコード件数]




orders 表と customer 表にそれぞれ custkey 索引が備わっており、orders 表では 1 つの custkey に対して 10 レコード存在するときに、以下 SQL の処理時間と算出式を記載せよ。

orders表のcustkey索引では、custkey1つに10レコード存在
→1つのcustkeyに対して「orders表のレコード10件」を読込む必要
→orders表の索引アクセスでは、アクセスするレコード数が10倍




| 質問番号 | 顧客番号 |
|---|---|
| Q01 | C01 |
| Q02 | C01 |
| Q03 | C02 |

| 注文番号 | 顧客番号 | 商品番号 |
|---|---|---|
| 001 | C01 | A01 |
| 002 | C01 | A02 |
| 003 | C03 | A01 |

例)1つの顧客番号に2つの注文番号
→Q01,Q02のそれぞれで「2件のC01」を読込む
→1回の索引アクセスで0.0001[s/件]×2[件]の時間が必要

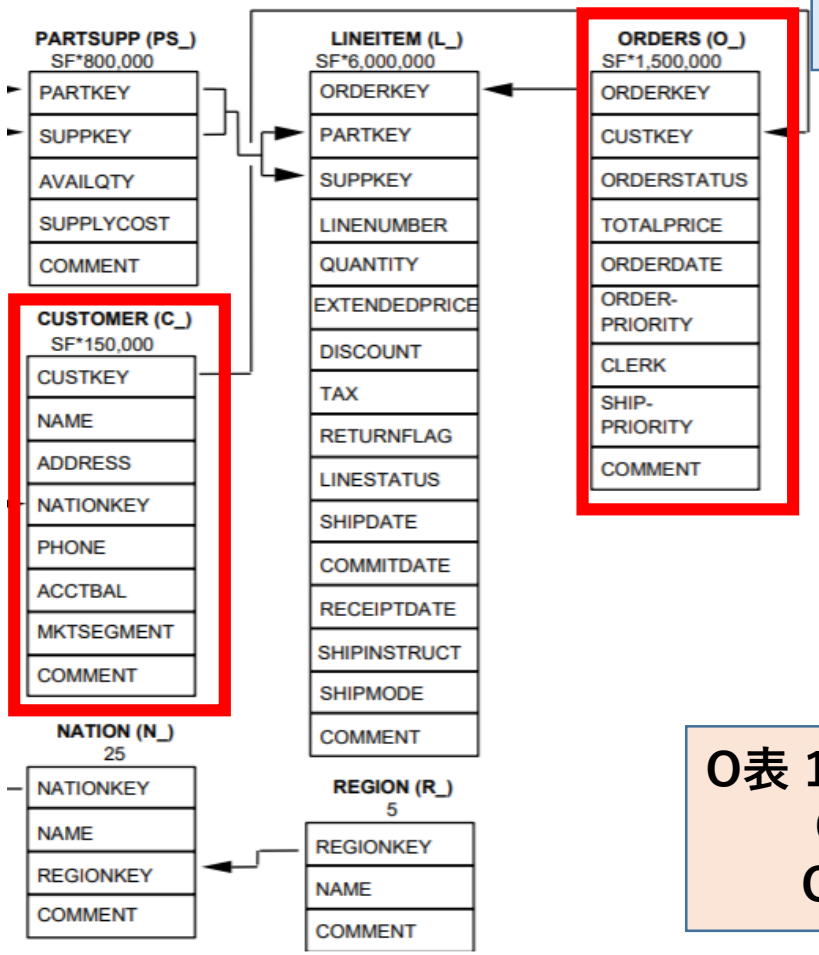
# 課題6-3
# (余裕あれば)
# 締切:11/23

orders 表と customer 表にそれぞれ custkey 索引が備わっており、orders 表では 1 つの custkey に対して 10 レコード存在するときに、以下 SQL の処理時間と算出式を記載せよ。

**ファンアウトが10**

d)select * from orders,customer where orders.custkey=customer.custkey 外表(駆動表)が customer 表

問d)ネストループ結合で、外部表がcustomer表なので
クエリ処理時間=外部表読込み時間+(外部表レコード×絞込み)×内部表読込み時間
=customer表の最短表読込み時間+(customer表のレコード数×1)×orders表の最短アクセス時間




**C表 1回の表スキャン:150,000[行]/1,000,000[行/GB]/0.5[GB/秒]=0.3[秒]**
**C表行数回のC表索引アクセス=150,000[行]*0.0001[秒]=15[秒]**
**→表スキャンが早い**

**150000[行]/1000000[行/GB]/0.5[GB/秒]**
**+150000[行]*10[ファンアウト]*0.0001[秒/行]=150.3秒**

**O表 1回の表スキャン:1,50,000[行]/1,000,000[行/GB]/0.5[GB/秒]=3[秒]**
**C表行数回のO表スキャン=150,000[行]*3[秒]=450,000[秒]**
**C表行数回のO表索引アクセス=150,000[行]*0.0001[秒]*10=150[秒]**

# 課題6 締切:11/23

余裕があれば6-4)下記は、ある牛丼屋の注文状況を記録している注文履歴テーブルである。次のデータを取得するためのSQLを作成してください。

```
create table order_t(o_date datetime,
                     order_id int not null,
                     detail_id int not null,
                     item_name varchar(20),
                     category char(1),
                     size char(1),
                     price int,
                     unit int,
                     total int, primary key (order_id,detail_id));
insert into order_t values('2022-11-01 10:10:10',1,1,'牛丼','F','S',2,400,800);
insert into order_t values('2022-11-01 10:10:10',1,2,'コーラ','D','S',1,200,200);
insert into order_t values('2022-11-01 10:10:10',1,3,'コーヒー','D','S',1,230,230);
insert into order_t values('2022-11-01 10:10:10',2,1,'牛丼','F','S',1,400,400);
 insert into order_t values('2022-11-01 10:10:10',2,2,'コーヒー','D','S',1,230,230);
select * from order_t;
```

**注文日時**

**一つの注文の中に、複数商品がある可能性**

**注文番号(主キー)、注文枝番(主キー)**

**商品名**

**分類(F:フード、D:ドリンク、O:その他)**

**サイズ(S:スモール、L:ラージ)**

a)注文順かつその明細順に、すべての注文データを取得する。

b)各注文について、ドリンク注文価格の総額を取得する。#注文を構成する全ての注文枝番について、分類Dなら集計

c)商品名毎の売り上げ集計を行い、売上額が大きい順に表示する。

# 課題6 締切:11/23

```
create table order_t(o_date datetime,
                order_id int not null,
                detail_id int not null,
                item_name varchar(20),
                category char(1),
                size char(1),
                price int,
                unit int,
                total int, primary key (order_id,detail_id));
insert into order_t values('2022-11-01 10:10:10',1,1,'牛丼','F','S',2,400,800);
insert into order_t values('2022-11-01 10:10:10',1,2,'コーラ','D','S',1,200,200);
insert into order_t values('2022-11-01 10:10:10',1,3,'コーヒー','D','S',1,230,230);
insert into order_t values('2022-11-01 10:10:10',2,1,'牛丼','F','S',1,400,400);
 insert into order_t values('2022-11-01 10:10:10',2,2,'コーヒー','D','S',1,230,230);
```

**注文日時**

**一つの注文の中に、複数商品がある可能性**

**注文番号(主キー)、注文枝番(主キー)**

**商品名**

**分類(F:フード、D:ドリンク、O:その他)**

**サイズ(S:スモール、L:ラージ)**

**select * from order_t order by order_id, detail_id;**

a)注文順かつその明細順に、すべての注文データを取得する。

**select order_id, sum(total) as sum_total from order_t where category='D' group by order_id**

b)各注文について、ドリンク注文価格の総額を取得する。#注文を構成する全ての注文枝番について、分類Dなら集計
c)商品名毎の売り上げ集計を行い、売上額が大きい順に表示する。

**select item_name, sum(total) from order_t group by item_name order by sum(total) desc;**