『日本語話し言葉コーパス』コアRDB版(CSJ-RDB)
Version 2.0
利用の手引き

第1部
CSJ-RDBの概要

CSJ-RDB

- CSJ-RDB とは
  - ✓ 『日本語話し言葉コーパス』(CSJ)のうちコア(201講演、約45時間)を対象
  - ✓ CSJ本体のXML文書に含まれる情報をほぼ反映(若干、追加・修正)
- 参考資料
  - ✓ CSJ-RDBの概要
    http://www.ninjal.ac.jp/corpus_center/csj/data/rdb-str/
  - ✓ CSJ-RDBの構成
    http://www.ninjal.ac.jp/corpus_center/csj/data/rdb-str/
  - ✓ CSJ本体との主な相違点
    http://www.ninjal.ac.jp/corpus_center/csj/data/rdb-diff/
  - ✓ CSJに付与されている各種ラベリングのマニュアル
    http://www.ninjal.ac.jp/corpus_center/csj/doc/

RDBとは

- 相互に関係づけ可能な複数のテーブルの集合

共通キー

Table1　談話情報

TalkID	TalkType	SpeakerID
A01F0055	monolog	459
D01F0023	dialog	514

共通キー

Table2　話者情報

SpeakerID	性別	出身地
459	男	東京都
514	男	神奈川県

Table3　単語情報

TalkID	単語	品詞
A01F0055	発表	名詞
A01F0055	し	動詞
A01F0055	まし	助動詞
A01F0055	た	助動詞
D01F0023	報告	名詞
D01F0023	し	動詞
D01F0023	ます	助動詞

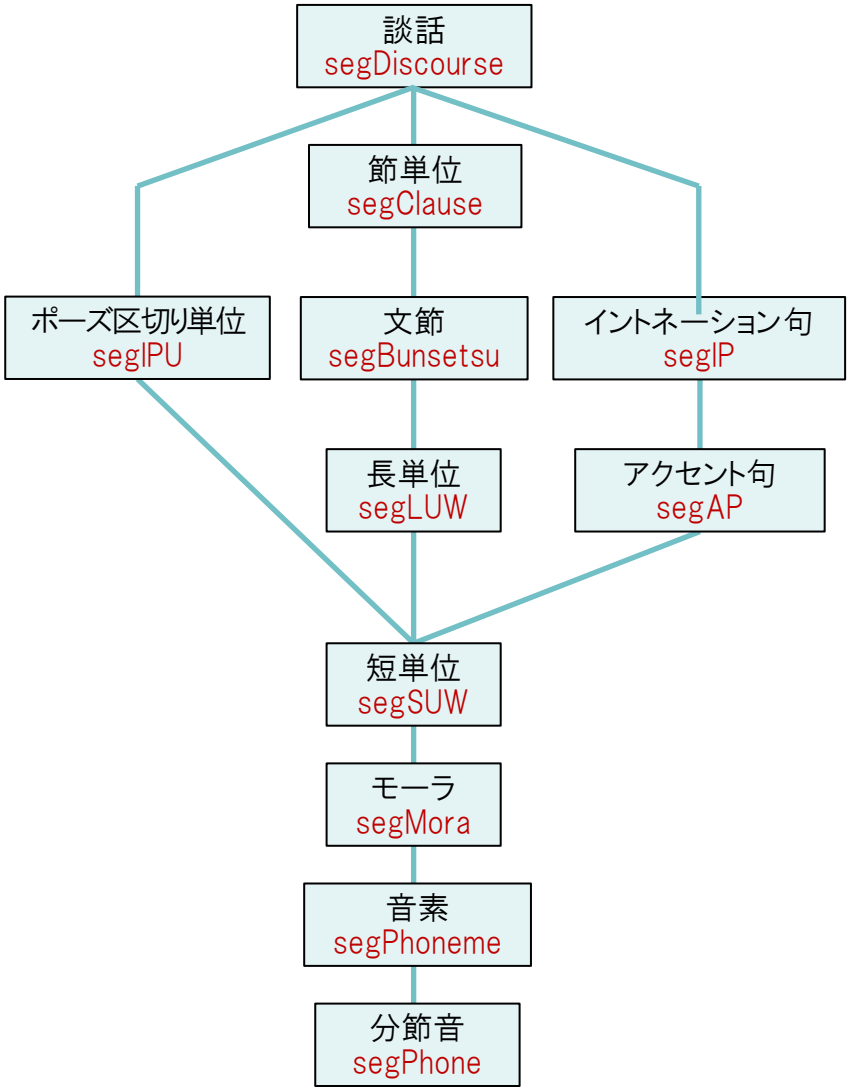
Table1とTable2を共通キー”SpeakerID“で、Table1とTable3を共通キー”TalkID”で関係づけて結合すると…

3つのテーブルを結合して作られたテーブル

TalkID	単語	品詞	TalkType	SpeakerID	性別	出身地
A01F0055	発表	名詞	monolog	459	男	東京都
A01F0055	し	動詞	monolog	459	男	東京都
A01F0055	まし	助動詞	monolog	459	男	東京都
A01F0055	た	助動詞	monolog	459	男	東京都
D01F0023	報告	名詞	dialog	514	男	神奈川県
D01F0023	し	動詞	dialog	514	男	神奈川県
D01F0023	ます	助動詞	dialog	514	男	神奈川県

CSJ-RDBの基本構成

- CSJコアには、形態論情報・節単位情報・係り受け情報・分節音情報・韻律情報など多様な情報が付与されている
- これらの情報を、談話中の要素を記述する複数の単位(＝セグメント、例：文節)と、単位間の関係を記述するリンクによって、各アノテーションを表現
- 各単位は右図のように層化
- 単位ごとに別々のテーブルで関連情報が表現
  ⇒セグメント･テーブル
- 節単位と短単位のように、親子関係にある２つの単位間の対応関係も、それぞれテーブルの形式で表現
  ⇒親子関係テーブル



セグメントテーブルのサンプル
―分節音(segPhone)―

ファイルID 分節音ID 開始時刻 終了時刻 話者情報 音素内容 音素クラス 無声化有無

TalkID	PhoneID	StartTime	EndTime	Channel	PhoneEntity	PhoneClass	Devoiced	…
A01F0055	05551385L	5.551385	5.632454	L	h	consonant	0	
A01F0055	05632454L	5.632454	5.698874	L	a	vowel	0	
A01F0055	05698874L	5.698874	5.760029	L	Q	special	0	
A01F0055	05760029L	5.760029	5.821184	L	SclS	others	0	
A01F0055	05821184L	5.821184	5.837566	L	py	consonant	0	
A01F0055	05837566L	5.837566	5.907904	L	o	vowel	0	
A01F0055	05907903L	5.907903	5.978241	L	H	special	0	
A01F0055	05978241L	5.978241	6.028152	L	sj	consonant	0	
A01F0055	06028153L	6.028153	6.078064	L	i	vowel	1	
A01F0055	06078064L	6.078064	6.16653	L	m	consonant	0	
A01F0055	06166530L	6.16653	6.277931	L	a	vowel	0	
A01F0055	06277931L	6.277931	6.382126	L	s	consonant	0	
A01F0055	06382126L	6.382126	6.532801	L	u	vowel	0	

全てのセグメント・テーブルに共通の列名

列名は異なるが全てのテーブルに存在

各テーブル固有の情報

親子関係テーブルのサンプル
―節単位(segClause)と文節(segBunsetsu)―

【セグメント・テーブル　節単位】

TalkID	ClauseID	StartTime	EndTime	Channel	Text	ClauseBoundaryLabel	(略)
A01F0067	00262895L	262.895042	264.895345	L	次の三つの課題を行ないました	[文末]	(略)

【セグメント・テーブル　文節】

TalkID	BunsetsuID	StartTime	EndTime	Channel	Text
A01F0067	00262895L	262.895042	263.240038	L	次の
A01F0067	00263240L	263.240038	263.769447	L	三つの
A01F0067	00263769L	263.769447	264.153538	L	課題を
A01F0067	00264154L	264.153538	264.895345	L	行ないました

【親子関係テーブル　節単位と文節】

TalkID	BunsetsuID	ClauseID	nth	len
A01F0067	00262895L	00262895L	1	4
A01F0067	00263240L	00262895L	2	4
A01F0067	00263769L	00262895L	3	4
A01F0067	00264154L	00262895L	4	4
ファイルID	子供の単位ID	親の単位ID	親の中の子供の位置	親の中の子供の数

第2部
RDBの操作環境

SQLite操作ツール

- 有償のソフトウウェア
  - Navicat　※1ヶ月間無償で利用可

    http://www.navicat.jp/store.html
    - ✓SQLite　（SQLiteのみ）
    - ✓Premium（各種DBに接続可）

    ※詳細は「Navicat使い方.pdf」をご覧ください

- 無償のソフトウェア（クエリ文の作成を補助するGUI操作なし）
  - DB Browser for SQLite

    http://sqlitebrowser.org/

    ※詳細は「DB-Browser使い方.pdf」をご覧ください

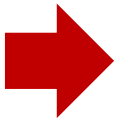
第3部
SQLクエリの基本

基本操作① SELECT文～列の選択～

- テーブルから指定した**列**を選択

テーブル名：infoTalk

TalkID	TalkType	SpeakerID
A01F0055	monolog	459
A02M0098	monolog	130
D01F0023	dialog	514
D01M0009	dialog	471



新しいテーブル

TalkID	TalkType
A01F0055	monolog
A02M0098	monolog
D01F0023	dialog
D01M0009	dialog

テーブルから“TalkID”と“TalkType”の列を選択

基本操作② WHERE句～行の抽出～

- テーブルから条件に合致した行を抽出

テーブル名：infoTalk

TalkID	TalkType	SpeakerID
A01F0055	monolog	459
A02M0098	monolog	130
D01F0023	dialog	514
D01M0009	dialog	471

新しいテーブル

TalkID	TalkType	SpeakerID
D01F0023	dialog	514
D01M0009	dialog	471

テーブルから、列“TalkType”が“dialog”である、という条件に合致した行を抽出

基本操作③ JOIN句～結合～

- 複数のテーブルを共通するキーにより関係づけて結合し、新しい一つのテーブルを作成

共通するキー

infoTalk

TalkID	TalkType	SpeakerID
A01F0055	monolog	459
A02M0098	monolog	130
D01F0023	dialog	514
D01M0009	dialog	471

infoSpeaker

SpeakerID	SpeakerSex	SpeakerBirthPlace
459	男	東京都
130	女	東京都
514	男	神奈川県
471	女	神奈川県

新しいテーブル

TalkID	TalkType	SpeakerID	SpeakerSex	SpeakerBirthPlace
A01F0055	monolog	459	男	東京都
A02M0098	monolog	130	女	東京都
D01F0023	dialog	514	男	神奈川県
D01M0009	dialog	471	女	神奈川県

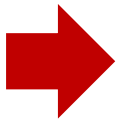
第3部　SQLクエリの基本
① SELECT文

SELECT文　～列の選択～

- テーブルから指定した**列**を選択

テーブル名：infoTalk

TalkID	TalkType	SpeakerID
A01F0055	monolog	459
A02M0098	monolog	130
D01F0023	dialog	514
D01M0009	dialog	471



新しいテーブル

TalkID	TalkType
A01F0055	monolog
A02M0098	monolog
D01F0023	dialog
D01M0009	dialog

テーブルから“TalkID”と“TalkType”の列を選択

※SQLで書くと

```
SELECT infoTalk.TalkID, infoTalk.TalkType FROM infoTalk
```

選択する列名（テーブル名.列名の形式）　テーブル名

SELECT文　～列の選択～

- 基本操作

```
SELECT テーブル名.列名1, テーブル名.列名2, …
FROM テーブル名
```

- テーブルが一つの場合は列名の前のテーブル名は省略可

```
SELECT 列名1, 列名2, …
FROM テーブル名
```

- *（半角アスタリスク）ですべての列を選択

```
SELECT　*
FROM テーブル名
```

例題

例題1　短単位テーブル(segSUW)から開始時間(StartTime)の列を選択

```
SELECT  segSUW.StartTime  FROM  segSUW
SELECT  StartTime  FROM  segSUW
```

← 列名は省略可

例題2　短単位テーブルから、開始時間、終了時間(EndTime)、テキスト(Text)の列を選択

```
SELECT  StartTime, EndTime, Text  FROM  segSUW
```

SELECT文　～列の演算～

- 選択する列同士を四則演算(+ － ＊ /)で計算することができる

```
SELECT　列名1 - 列名2
FROM　テーブル名
```

```
SELECT　( 列名1 + 列名2 ) / 列名3
FROM　テーブル名
```

SELECT文　～列に別名定義～

- ASを用いて列や計算結果に別名（エイリアス）をつけることができる

```
SELECT　 列名1 - 列名2 AS 名前
FROM　テーブル名
```

例題

例題3　短単位テーブルを対象に、各短単位の継続時間長を算出し、列名を「継続時間」とした上で、短単位のテキストと合わせて表示

```
SELECT （EndTime － StartTime） AS 継続時間, Text FROM segSUW
```

第3部 SQLクエリの基本
② WHERE句

WHERE 句　～行の抽出～

- テーブルから条件に合致した行を抽出

テーブル名：infoTalk

TalkID	TalkType	SpeakerID
A01F0055	monolog	459
A02M0098	monolog	130
D01F0023	dialog	514
D01M0009	dialog	471

新しいテーブル

TalkID	TalkType	SpeakerID
D01F0023	dialog	514
D01M0009	dialog	471

テーブルから、列“TalkType”が”dialog”である、という条件に合致した行を抽出

※SQLで書くと

```
SELECT infoTalk.* FROM infoTalk WHERE infoTalk.TalkType = "dialog"
```

抽出する列名（「*」は全ての列）　テーブル名　抽出条件（列“TalkType”が“dialog”である行）

WHERE 句　～比較演算子～

- **基本操作**

```
SELECT 列名1,列名2 …
FROM　 テーブル名
WHERE　条件式
```

- **条件式で用いる演算子1**　比較演算子

演算子	使用例	説明	補足
=	a = b	a と b は等しい	「==」でも可
<>	a <> b	a と b は等しくない	「!=」でも可
>	a > b	a は b より大きい	
>=	a >= b	a は b 以上	
<	a < b	a は b より小さい	
<=	a <= b	a は b 以下	

例）
```
WHERE　StartTime　=　60　　　# 開始時間が60秒に一致
WHERE　TalkID　=　“D01F0023”　# ファイルIDが“D01F0023”に一致
WHERE　StartTime　>= 400　　# 開始時間が400秒以降
```

例題

例題4　モーラテーブル(segMora)からアクセント核の有無(PerceivedAcc)が 1 である(アクセント核がある)行を抽出し、全ての列(*)を選択

```
SELECT * FROM segMora WHERE PerceivedAcc = 1
```

例題5　モーラテーブルのモーラ記号(moraEntity)が「ア」である行を抽出し、全ての 列を選択

```
SELECT * FROM segMora WHERE MoraEntity = "ア"
```

※文字列の場合は引用符で囲む

例題6　モーラテーブルから継続長が 0.1(秒)より大きい行を抽出し、全ての列を選択

```
SELECT * FROM segMora WHERE EndTime - StartTime > 0.1
```

WHERE 句 ～パターンマッチング～

条件式で用いる演算子2　パターンマッチング

［形式］ 列名　LIKE　“パターン”　…列名がパターンにマッチする場合
列名　NOT LIKE　“パターン”　…列名がパターンにマッチしない場合

使用できるワイルドカード

ワイルドカード	意味
%	任意の0文字以上の文字列
_	任意の1文字

例）
WHERE　Text　LIKE　“あ%”　…「あ＋ 0文字以上の任意の文字列」の形式のテキスト
（「あ」「あか」「あかい」など）
WHERE　Text　LIKE　“%い”　…「0文字以上の任意の文字列＋い」の形式のテキスト
（「い」「はい」「美しい」「大きい」「よろしい」など）
WHERE　Text　LIKE　“あ_”　…「あ＋ 任意の1文字」の形式のテキスト
（「ああ」「あか」「あさ」「あほ」など）

例題

例題7　文節テーブル(segBunsetsu)から、テキスト(Text)が「は」で終了する行を抽出し、全ての列を選択

```
SELECT * FROM segBunsetsu WHERE Text LIKE "%は"
```

例題8　文節テーブルから、テキストがフィラーである行を抽出し、全ての列を選択

```
SELECT * FROM segBunsetsu WHERE Text LIKE "(F %)"
```

※ フィラーは「(F あのー)」や「(F えー)」のように、記号 (F) で囲まれてる

WHERE 句　～論理演算子～

- **条件式で用いる演算子3**　論理演算子

演算子	使用例	説明
AND	p AND q	p と q が共に真の場合
OR	p OR q	p か q の少なくとも一つが真の場合
NOT	NOT p	p が真ではない（偽である）場合

pとqは条件式

例）WHERE　TalkID = ″D01M0009″ AND　Text　LIKE　″%です”

WHERE　MoraEntity = “ア”　OR　MoraEntity = “ウ″

WHERE　MoraEntity = “ア”　OR　MoraEntity = “ウ”　OR　MoraEntity = “オ″

例題

例題9 節単位テーブル(segClause)から、節境界ラベル(ClauseBoundaryLabel)が“/並列節シ/”である、**あるいは**、節境界ラベルが“/並列節デ/”である行を抽出し、全ての列を選択

```
SELECT  *  FROM  segClause
WHERE  ClauseBoundaryLabel  LIKE  "/並列節シ/"
   OR  ClauseBoundaryLabel  LIKE  "/並列節デ/"
```

例題10 節単位テーブルから、節境界ラベルが強境界の並列節に該当する行で、**かつ**、節境界ラベルが“/並列節シ/”でも“/並列節デ/”でもない行を抽出し全ての列を選択

```
SELECT  *  FROM  segClause
WHERE  ClauseBoundaryLabel  LIKE  "/並列節%/"
  AND ( ClauseBoundaryLabel  NOT LIKE  "/並列節デ/"
   OR  ClauseBoundaryLabel  NOT LIKE  "/並列節デ/" )
```

※ AND条件の方がOR条件より優先される。ORを優先させる場合はカッコでくくる

※ 並列節の強境界は”/並列節XXX/” (XXXは任意の文字列)の形式で表現

第3部 SQLクエリの基本
③ JOIN句　内部結合

INNER JOIN句

- 対応する値がある行のみ結合

■ infoTalk

対応させるキー

TalkID	TalkType	SpeakerID
A01F0055	monolog	459
A02M0098	monolog	130
D01F0023	dialog	514
D01M0009	dialog	471
R00M0187	monolog	423

対応

■ infoSpeaker

SpeakerID	SpeakerSex	SpeakerBirthPlace
459	男	東京都
130	女	東京都
514	男	神奈川県
471	女	神奈川県
131	男	群馬県

■ 新しいテーブル

TalkID	TalkType	SpeakerID	SpeakerSex	SpeakerBirthPlace
A01F0055	monolog	459	男	東京都
A02M0098	monolog	130	女	東京都
D01F0023	dialog	514	男	神奈川県
D01M0009	dialog	471	女	神奈川県

INNER JOIN句

```
SELECT  infoTalk.*,  infoSpeaker.SpeakerSex,  infoSpeaker.SpeakerBirthPlace
FROM  infoTalk
INNER JOIN  infoSpeaker  ON  InfoTalk.SpeakerID  =  infoSpeaker.SpeakerID
```

結合元のテーブル名

結合先のテーブル名

対応させるキー

結合元
■ infoTalk

対応させるキー

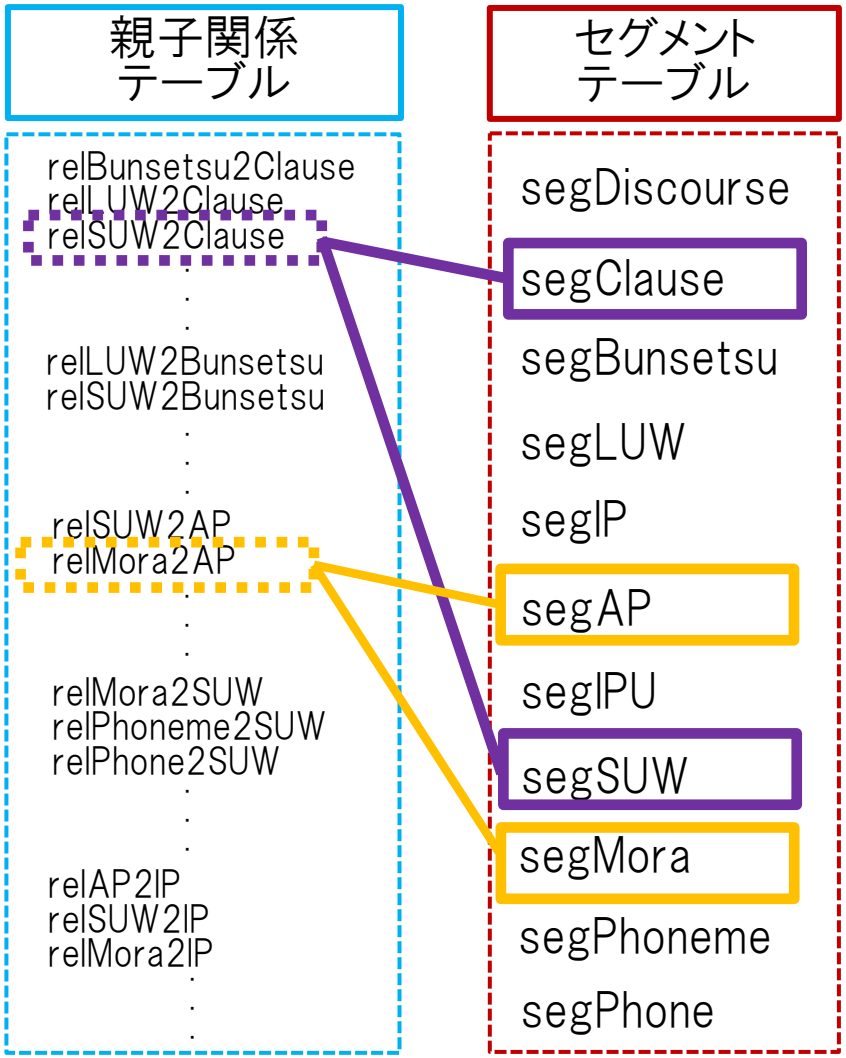
結合先
■ infoSpeaker

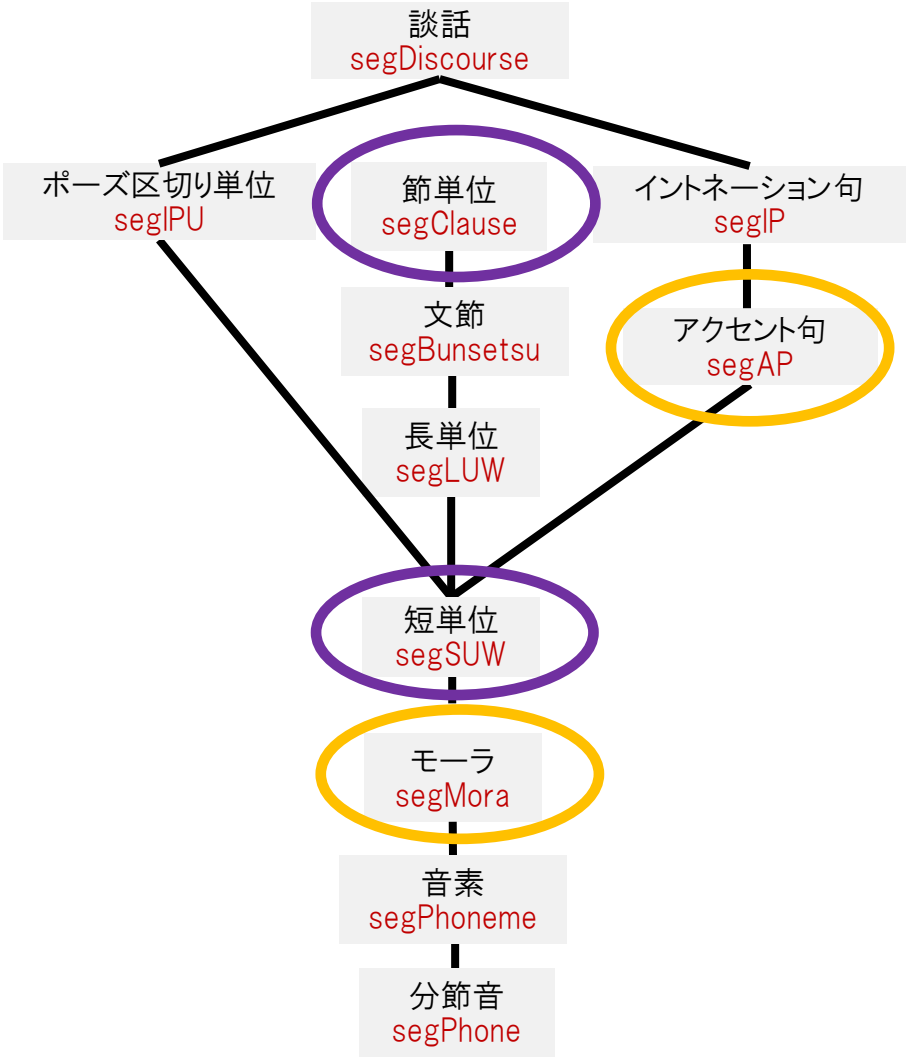
TalkID	TalkType	SpeakerID
A01F0055	monolog	459
A02M0098	monolog	130
D01F0023	dialog	514
D01M0009	dialog	471
R00M0187	monolog	423

SpeakerID	SpeakerSex	SpeakerBirthPlace
459	男	東京都
130	女	東京都
514	男	神奈川県
471	女	神奈川県
131	男	群馬県

CSJ-RDBの構成　―親子関係テーブル―

親子関係テーブル：親子関係にある２つのセグメント間の関係を表現したテーブル
(セグメントテーブル同士を結合する際などに用いる)





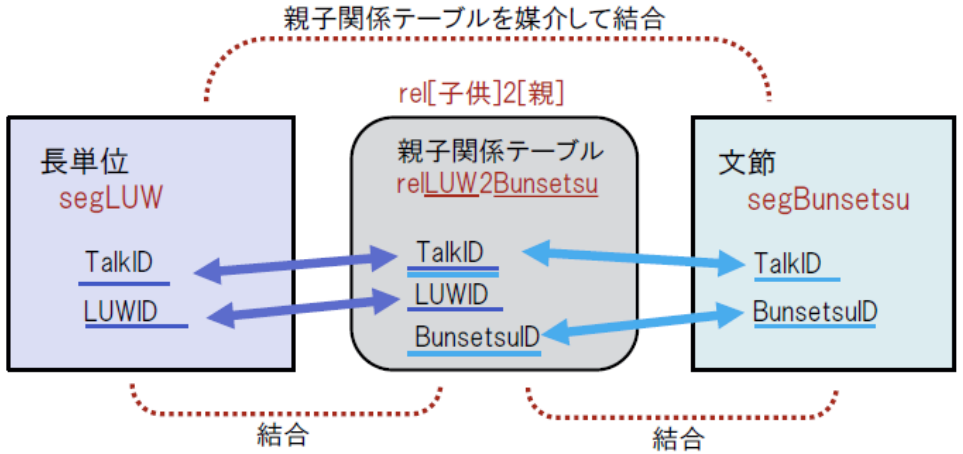
親子関係テーブルの具体例

対応させるキー

子供の単位ID

■ segLUW （子供のテーブル）

TalkID	LUWID	Text
A01F0055	00053773L	いつ頃
A01F0055	00054186L	から
A01F0055	00054505L	可能
A01F0055	00054862L	な
A01F0055	00054980L	のでしょ
A01F0055	00055415L	う
A01F0055	00055478L	か



■ relLUW2Bunsetsu （親子関係テーブル）

TalkID	LUWID	BunsetsuID	nth	len
A01F0055	00053773L	00053773L	1	2
A01F0055	00054186L	00053773L	2	2
A01F0055	00054505L	00054505L	1	5
A01F0055	00054862L	00054505L	2	5
A01F0055	00054980L	00054505L	3	5
A01F0055	00055415L	00054505L	4	5
A01F0055	00055478L	00054505L	5	5

対応させるキー

親の単位ID

■ segBunsetsu （親のテーブル）

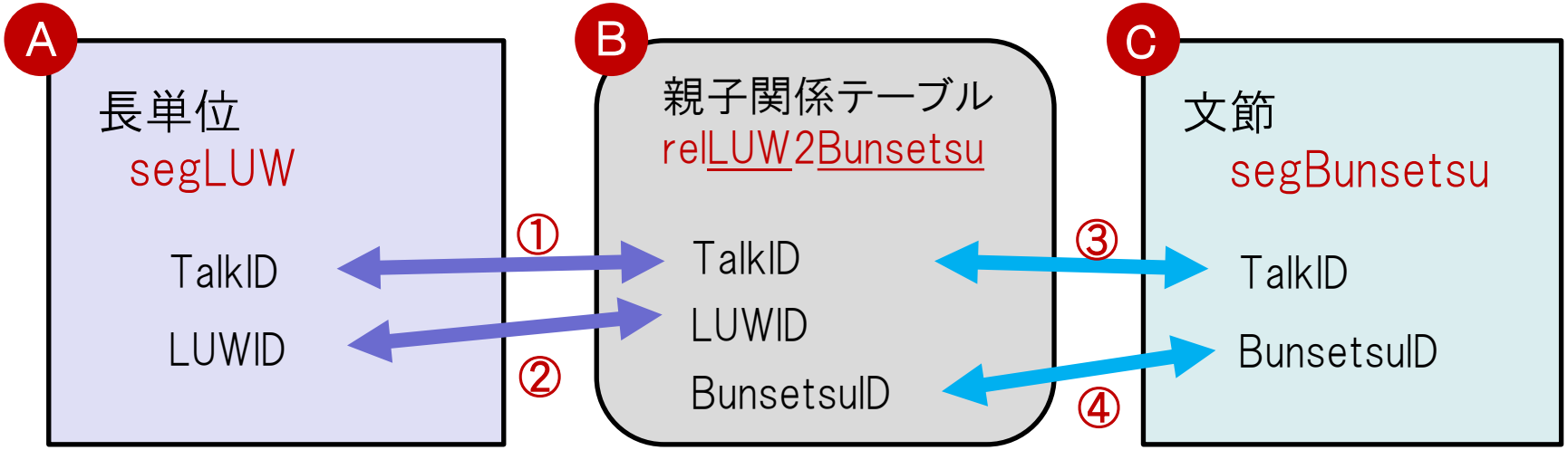
TalkID	BunsetsuID	Text
A01F0055	00053773L	いつ頃から
A01F0055	00054505L	可能なのでしょうか

■ 子供（長単位）と親（文節）を結合した後のテーブル

TalkID	nth	len	LUWID	Text1	BunsetsuID	Text2
A01F0055	1	2	00053773L	いつ頃	00053773L	いつ頃から
A01F0055	2	2	00054186L	から	00053773L	いつ頃から
A01F0055	1	5	00054505L	可能	00054505L	可能なのでしょうか
A01F0055	2	5	00054862L	な	00054505L	可能なのでしょうか
A01F0055	3	5	00054980L	のでしょ	00054505L	可能なのでしょうか
A01F0055	4	5	00055415L	う	00054505L	可能なのでしょうか
A01F0055	5	5	00055478L	か	00054505L	可能なのでしょうか

親の単位（文節）は子供の単位（文節）の数だけ繰り返されることに注意

SQL文　～長単位と節単位の結合の場合～




```
SELECT  segLUW.*,  segBunsetsu.*
FROM  segLUW        A
INNER JOIN   relLUW2Bunsetsu        B
        ON  segLUW.TalkID = relLUW2Bunsetsu.TalkID                  ①
       AND  segLUW.LUWID = relLUW2Bunsetsu.LUWID                    ②
INNER JOIN   segBunsetsu        C
        ON  segBunsetsu.TalkID = relLUW2Bunsetsu.TalkID             ③
       AND segBunsetsu.BunsetsuID = relLUW2Bunsetsu.BunsetsuID      ④
```

例題

例題11　文節テーブルと節単位テーブルを結合し、テキストが「は」で終了する文節であり、かつ節境界ラベルが“/並列節ガ/”であるものを抽出し、TalkID、文節のテキスト、節単位のテキストの列を選択

```
   SELECT  segBunsetsu.Text,  segClause.Text  FROM  segBunsetsu
INNER JOIN  relBunsetsu2Clause
        ON  segBunsetsu.TalkID = relBunsetsu2Clause.TalkID
       AND  segBunsetsu.BunsetsuID = relBunsetsu2Clause.BunsetsuID
INNER JOIN  segClause
        ON  segClause.TalkID = relBunsetsu2Clause.TalkID
       AND  segClause. ClauseID = relBunsetsu2Clause. ClauseID
     WHERE  segBunsetsu.Text  LIKE  "%は"
       AND  segClause. ClauseBoundaryLabel  = "/並列節ガ/"
```

⇒ このように、二つ以上の単位に関わる情報を知りたい場合には、親子関係テーブルを利用して複数の単位を結合する

親子関係テーブルの nth と len の活用

■ segLUW （子供のテーブル）

TalkID	LUWID	Text
A01F0055	00053773L	いつ頃
A01F0055	00054186L	から
A01F0055	00054505L	可能
A01F0055	00054862L	な
A01F0055	00054980L	のでしょ
A01F0055	00055415L	う
A01F0055	00055478L	か

■ relLUW2Bunsetsu （親子関係テーブル）

TalkID	LUWID	BunsetsuID	nth	len
A01F0055	00053773L	00053773L	1	2
A01F0055	00054186L	00053773L	2	2
A01F0055	00054505L	00054505L	1	5
A01F0055	00054862L	00054505L	2	5
A01F0055	00054980L	00054505L	3	5
A01F0055	00055415L	00054505L	4	5
A01F0055	00055478L	00054505L	5	5

親単位中の子供単位の数（len）と位置（nth=何番目の子供か）

この例：文節(親)「可能なのでしょうか」には5つの子供(文節)が含まれており、このうち文節「な」は2番目の子供

■ segBunsetsu （親のテーブル）

TalkID	BunsetsuID	Text
A01F0055	00053773L	いつ頃から
A01F0055	00054505L	可能なのでしょうか

■ 子供(長単位)と親(文節)を結合した後のテーブル

TalkID	nth	len	LUWID	Text1	BunsetsuID	Text2
A01F0055	1	2	00053773L	いつ頃	00053773L	いつ頃から
A01F0055	2	2	00054186L	から	00053773L	いつ頃から
A01F0055	1	5	00054505L	可能	00054505L	可能なのでしょうか
A01F0055	2	5	00054862L	な	00054505L	可能なのでしょうか
A01F0055	3	5	00054980L	のでしょ	00054505L	可能なのでしょうか
A01F0055	4	5	00055415L	う	00054505L	可能なのでしょうか
A01F0055	5	5	00055478L	か	00054505L	可能なのでしょうか

例題

例題12　節単位の先頭の文節を抽出し、TalkID、文節のテキスト、節単位のテキストを選択

```
SELECT   segBunsetsu.Text,    segClause.Text
  FROM   segBunsetsu
INNER JOIN   relBunsetsu2Clause
    ON   segBunsetsu.TalkID = relBunsetsu2Clause.TalkID
   AND   segBunsetsu.BunsetsuID = relBunsetsu2Clause.BunsetsuID
INNER JOIN   segClause
    ON   segClause.TalkID = relBunsetsu2Clause.TalkID
   AND   segClause. ClauseID = relBunsetsu2Clause. ClauseID
 WHERE   relBunsetsu2Clause.nth  = 1
```

※ 節単位(親)の先頭の文節(子供)を指定するには、
親子関係テーブルの nth が 1という条件を設定すればよい。
前ページの親子関係テーブルを確認のこと。

例題

例題13　節単位の末尾の文節を抽出し、TalkID、文節のテキスト、節単位のテキストを選択

```
SELECT   segBunsetsu.Text,    segClause.Text
  FROM   segBunsetsu
INNER JOIN   relBunsetsu2Clause
    ON   segBunsetsu.TalkID = relBunsetsu2Clause.TalkID
   AND   segBunsetsu.BunsetsuID = relBunsetsu2Clause.BunsetsuID
INNER JOIN   segClause
    ON   segClause.TalkID = relBunsetsu2Clause.TalkID
   AND   segClause. ClauseID = relBunsetsu2Clause. ClauseID
 WHERE   relBunsetsu2Clause.nth  = relBunsetsu2Clause.len
```

※ 節単位(親)の最後の文節(子供)を指定するには、
親子関係テーブルの nth が len と一致するという条件を設定すればよい。
前々ページの親子関係テーブルを確認のこと。

例題

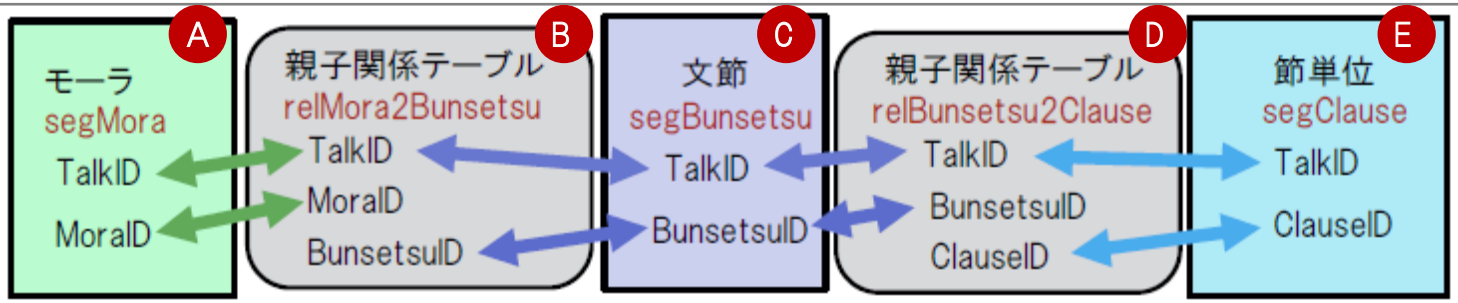
例題14 節単位の最後から2番目の文節であり、かつ、テキストが「は」で終了する文節を抽出し、各文節の継続長を計算して“Duration”という別名を付けた上で、TalkID、節単位のテキスト、文節のテキストと合わせて表示。

```
SELECT  segClause.TalkID, segClause.Text, segBunsetsu.Text,
        segBunsetsu.EndTime - segBunsetsu.StartTime  AS  Duration
FROM    segBunsetsu
INNER JOIN  relBunsetsu2Clause
ON      segBunsetsu.TalkID = relBunsetsu2Clause.TalkID
AND     segBunsetsu.BunsetsuID = relBunsetsu2Clause.BunsetsuID
INNER JOIN  segClause
ON      segClause.TalkID = relBunsetsu2Clause.TalkID
AND     segClause. ClauseID = relBunsetsu2Clause. ClauseID
WHERE   relBunsetsu2Clause.nth  = relBunsetsu2Clause.len  -  1
AND     segBunsetsu.Text    LIKE  ″%は″
```

例題

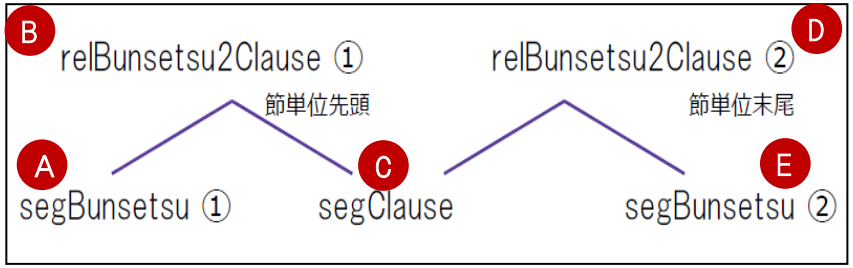
例題15 「モーラテーブル」「文節テーブル」「節単位テーブル」の3つのテーブルを結合し、TalkID、モーラ記号、文節のテキスト、節単位のテキストの列を選択

```
SELECT  segMora.TalkID,  segMora.MoraEntity,  segBunsetsu.Text, segClause.Text
  FROM  segMora      A
INNER JOIN  relMora2Bunsetsu    B
    ON  segMora.TalkID = relMora2Bunsetsu.TalkID
   AND  segMora.MoraID = relMora2Bunsetsu.MoraID
INNER JOIN   segBunsetsu      C
    ON  segBunsetsu.TalkID = relMora2Bunsetsu.TalkID
   AND  segBunsetsu.BunsetsuID = relMora2Bunsetsu.BunsetsuID
INNER JOIN   relBunsetsu2Clause     D
    ON   segBunsetsu.TalkID = relBunsetsu2Clause.TalkID
   AND   segBunsetsu.BunsetsuID = relBunsetsu2Clause.BunsetsuID
INNER JOIN   segClause   E
    ON   segClause.TalkID = relBunsetsu2Clause.TalkID
   AND   segClause. ClauseID = relBunsetsu2Clause. ClauseID
```



例題

例題16　節単位の先頭の文節のテキストが「は」で終わり、かつ、
節単位の末尾の文節のテキストが「思」で始まる行を抽出し、
TalkID、節単位のテキスト、節単位の先頭と末尾の文節のテキストを表示



左図のように、同じ節を共有する2つの文節（segBunsetsu①、②）を、それぞれ2つの親子関係テーブル（relBunsetsu2Clause①、②）で結合し、①の条件を「節の先頭」かつ「は」で終了、②の条件を「節の末尾」かつ「思」で開始、と設定する

```
SELECT segClause.TalkID, segClause.Text, segB1.Text, segB2.Text
  FROM segBunsetsu AS segB1                                    (A)
INNER JOIN relBunsetsu2Clause AS rel1                          (B)
    ON segB1.TalkID = rel1.TalkID
   AND segB1.BunsetsuID = rel1.BunsetsuID
INNER JOIN segClause                                           (C)
    ON rel1.TalkID = segClause.TalkID
   AND rel1.ClauseID = segClause.ClauseID
INNER JOIN relBunsetsu2Clause AS rel2                          (D)
    ON segClause.TalkID = rel2.TalkID
   AND segClause.ClauseID = rel2.ClauseID
INNER JOIN segBunsetsu AS segB2                                (E)
    ON rel2.TalkID = segB2.TalkID
   AND rel2.BunsetsuID = segB2.BunsetsuID
 WHERE rel1.nth = 1        AND segB1.Text LIKE "%は"           ①「節の先頭」かつ「は」で終了
   AND rel2.nth = rel2.len AND segB2.Text LIKE "思%"           ②「節の末尾」かつ「思」で開始
```

同じ名前のテーブルがあるためASで別名を付ける

例題

例題17　節単位内の隣接する二つの短単位のテキストの組合せを抽出し、TalkIDと合わせて表示

TalkID	SUWID	ClauseID	nth	len
D01F0023	00001006R	00001006R	1	5
D01F0023	00001467R	00001006R	2	5
D01F0023	00001545R	00001006R	3	5
D01F0023	00002012R	00001006R	4	5
D01F0023	00002173R	00001006R	5	5

relSUW2Clause ①

Text
よろしく
お
願い
し
ます

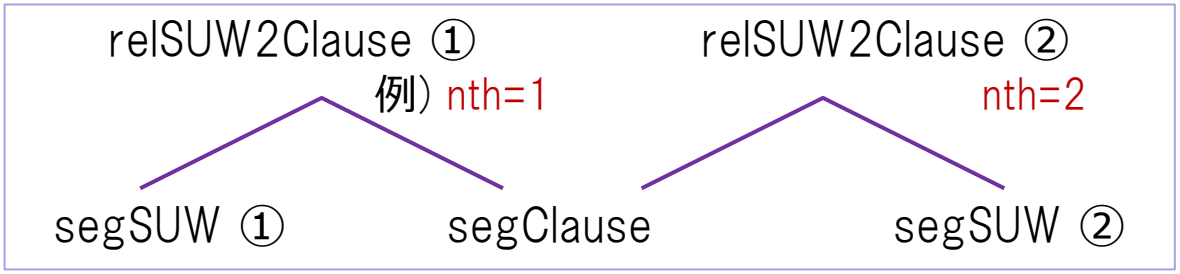
segSUW ①

Text
よろしく
お
願い
し
ます

segSUW ②

TalkID	SUWID	ClauseID	nth	len
D01F0023	00001006R	00001006R	1	5
D01F0023	00001467R	00001006R	2	5
D01F0023	00001545R	00001006R	3	5
D01F0023	00002012R	00001006R	4	5
D01F0023	00002173R	00001006R	5	5

relSUW2Clause ②



例題16と同様、同じ節単位を媒介して2つの短単位を結合。ただし、②の節単位中の位置（nth）は、①のn位置よりも1つ大きい。
この条件をWHERE句で指定する。

例題

例題17　節単位内の隣接する二つの短単位のテキストの組合せを抽出し、TalkIDと合わせて表示

```
   SELECT  segS1.TalkID, segS1.Text, segS2.Text
     FROM  segSUW  AS  segS1
INNER JOIN  relSUW2Clause  AS  rel1
        ON  segS1.TalkID  =  rel1.TalkID
       AND  rel1.SUWID  =  segS1.SUWID
INNER JOIN  segClause
        ON  rel1.TalkID  =  segClause.TalkID
       AND  rel1.ClauseID  =  segClause.ClauseID
INNER JOIN  relSUW2Clause  AS  rel2
        ON  segClause.TalkID  =  rel2.TalkID
       AND  rel2.ClauseID  =  segClause.ClauseID
INNER JOIN  segSUW  AS  segS2
        ON  rel2.TalkID  =  segS2.TalkID
       AND  segS2.SUWID  =  rel2.SUWID
    WHERE  rel1.nth  =  rel2.nth - 1
```

同じ名前のテーブルがあるためASで別名を付ける

①より②のnthの方が1大きい（rel1.nth + 1 = rel2.nth でも同じ）

例題

例題18 （談話全体で）隣接する二つの短単位のテキストの組合せを抽出し、TalkIDと合わせて表示

※ 例題17の「節」のかわりに「談話」（segDiscouse, relSUW2Discourse）を用いる

```
   SELECT  segS1.TalkID, segS1.Text, segS2.Text
     FROM  segSUW  AS  segS1
INNER JOIN  relSUW2Discourse  AS  rel1
        ON  segS1.TalkID  =  rel1.TalkID
       AND  rel1.SUWID  =  segS1.SUWID
INNER JOIN  segDiscourse
        ON  rel1.TalkID  =  segDiscourse.TalkID
       AND  rel1.DiscourseID  =  segDiscourse.DiscourseID
INNER JOIN  relSUW2Discourse  AS  rel2
        ON  segDiscourse.TalkID  =  rel2.TalkID
       AND  rel2.DiscourseID  =  segDiscourse.DiscourseID
INNER JOIN  segSUW  AS  segS2
        ON  rel2.TalkID  =  segS2.TalkID
       AND  segS2.SUWID  =  rel2.SUWID
    WHERE  rel1.nth  =  rel2.nth - 1
```

第3部 SQLクエリの基本
④ JOIN句　外部結合

内部結合　INNER JOIN　復習

- 対応する値がある行のみ結合

■ infoTalk

TalkID	TalkType	SpeakerID
A01F0055	monolog	459
A02M0098	monolog	130
D01F0023	dialog	514
R00M0187	monolog	423

出力されず

対応させるキー

対応

■ infoSpeaker

SpeakerID	SpeakerSex	SpeakerBirthPlace
459	男	東京都
130	女	東京都
514	男	神奈川県
131	男	群馬県

出力されず

■ 新しいテーブル

TalkID	TalkType	SpeakerID	SpeakerSex	SpeakerBirthPlace
A01F0055	monolog	459	男	東京都
A02M0098	monolog	130	女	東京都
D01F0023	dialog	514	男	神奈川県

外部結合 LEFT OUTER JOIN (= LEFT JOIN)

対応する列＋左側のテーブルにのみ存在する列も取得

■ infoTalk

TalkID	TalkType	SpeakerID
A01F0055	monolog	459
A02M0098	monolog	130
D01F0023	dialog	514
R00M0187	monolog	423

対応させるキー

対応

■ infoSpeaker

SpeakerID	SpeakerSex	SpeakerBirthPlace
459	男	東京都
130	女	東京都
514	男	神奈川県
131	男	群馬県

左側の InfoTalkテーブルにしか存在しないが…

右側の InfoSpeakerテーブルにしか存在しない列は取得されない

■ 新しいテーブル

TalkID	TalkType	SpeakerID	SpeakerSex	SpeakerBirthPlace
A01F0055	monolog	459	男	東京都
A02M0098	monolog	130	女	東京都
D01F0023	dialog	514	男	神奈川県
R00M0187	monolog	423	NULL	NULL

取得

infoSpeakerに関する列の値は空

外部結合　LEFT OUTER JOIN

```
SELECT  infoTalk.*,  infoSpeaker.SpeakerSex,  infoSpeaker.SpeakerBirthPlace

FROM  infoTalk
        └──────┘
        結合元(左)のテーブル名
LEFT OUTER JOIN  infoSpeaker  ON  InfoTalk.SpeakerID  =  infoSpeaker.SpeakerID
                      └──────┘                └──────────────────────────┘
                      結合先(右)のテーブル名              対応させるキー
```

結合元

■ infoTalk

TalkID	TalkType	SpeakerID
A01F0055	monolog	459
A02M0098	monolog	130
D01F0023	dialog	514
R00M0187	monolog	423

対応させるキー

結合先

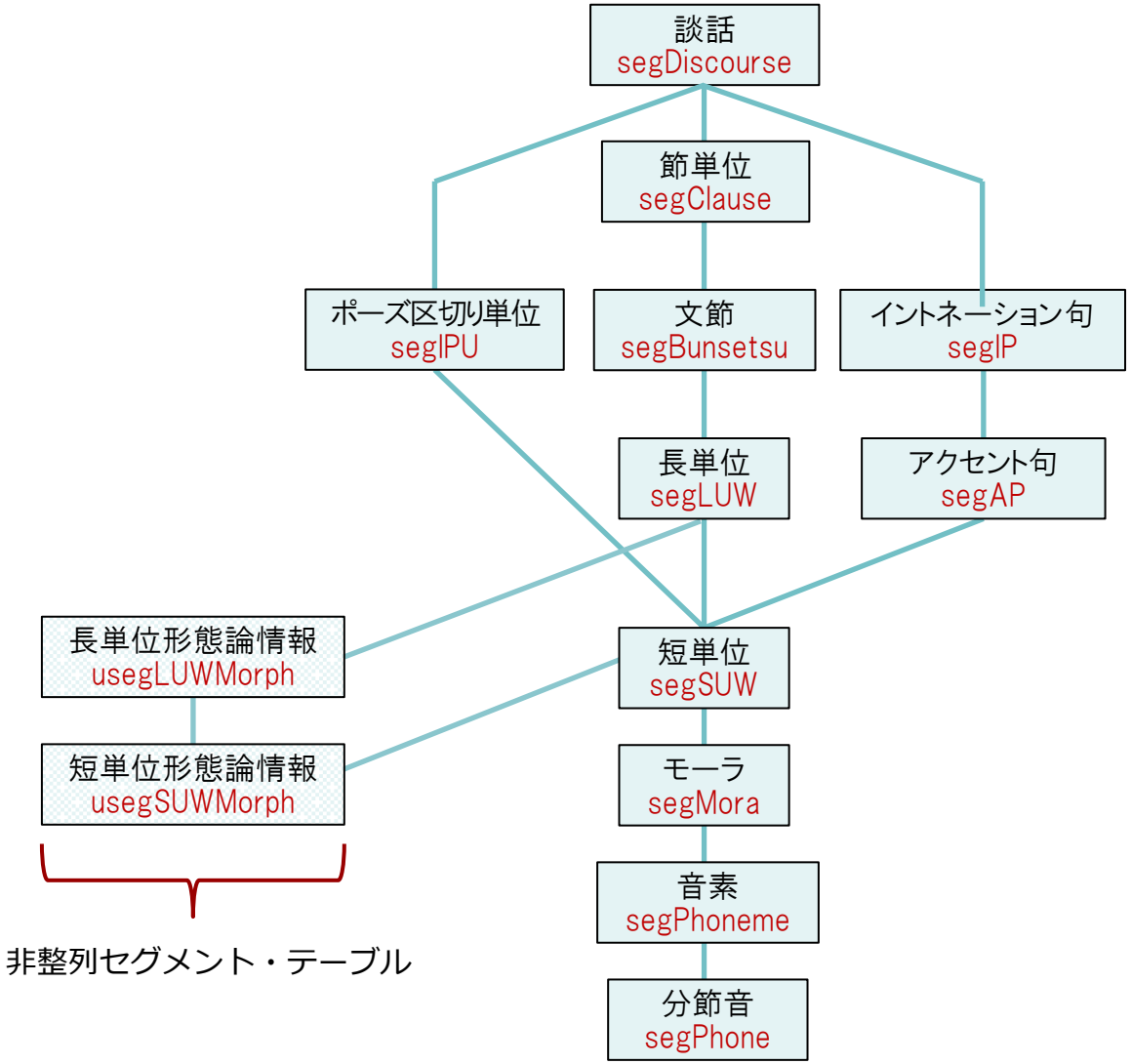
■ infoSpeaker

SpeakerID	SpeakerSex	SpeakerBirthPlace
459	男	東京都
130	女	東京都
514	男	神奈川県
131	男	群馬県

➡ 外部結合の例題は、例題22を参照

第4部
CSJ-RDBの構成 応用編

CSJ-RDBの構成 ―非整列セグメント・テーブル―



非整列セグメント・テーブル（短単位・長単位）

複数の語が融合して、分割できない一つの要素を形成する場合

■ セグメント･テーブル segSUW （短単位）

TalkID	SUWID	StartTime	EndTime	Text	word
A01F0122	00513758L	513.757614	514.114756	第一	(W daici)
A01F0122	00514115L	514.114756	514.483693	母音	bo'iN

■ 非整列セグメント･テーブル usegSUWMorph （短単位形態素情報）

TalkID	SUWID	nth	len	PlainOrthographicTranscription	SUWLemma	SUWPOS
A01F0122	00513758L	1	2	第	第	接頭辞
A01F0122	00513758L	2	2	一	一	名詞
A01F0122	00514115L	1	1	母音	母音	名詞

提供される親子関係テーブル：

usegSUWMorph（子供）と segSUW（親）の間
usegLUWMorph（子供）と segLUW（親）の間
usegSUWMorph（子供）と usegLUWMorph（親）の間

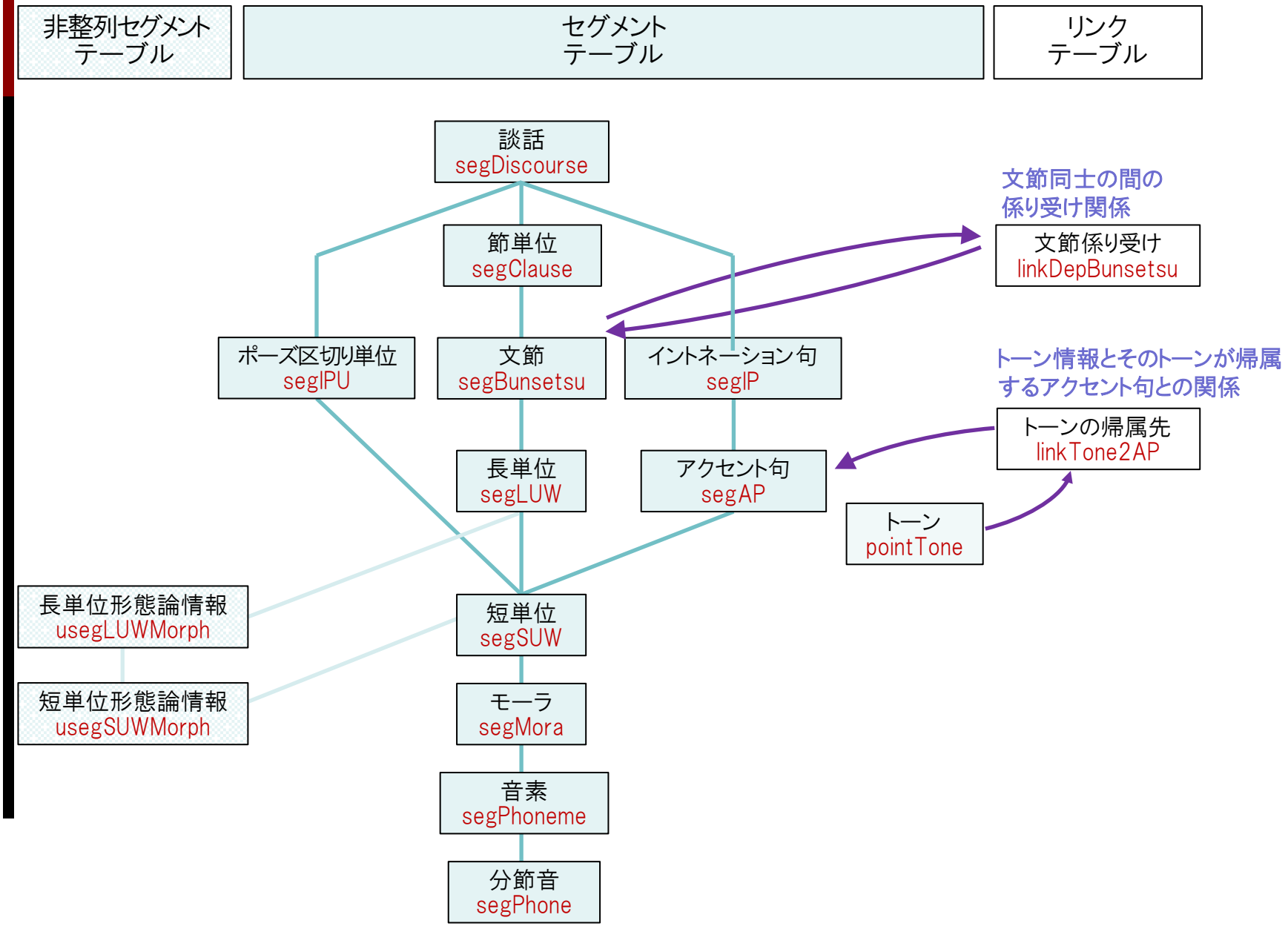
例題

例題19 文節の最後の短単位の品詞が助詞であるものを抽出し、文節のテキストと短単位の出現形（usegSUWMorph.OrthographicTranscription）を表示

※品詞情報のある短単位形態論情報と文節間の親子関係テーブルはないため、「Ⓐ短単位形態論情報⇔Ⓑ短単位⇔Ⓒ文節」の3つのテーブルを結合

```
SELECT  segBunsetsu.Text,  usegSUWMorph.OrthographicTranscription
  FROM  usegSUWMorph  Ⓐ
INNER JOIN  relSUWMorph2SUW      ← Ⓐ と Ⓑ の親子関係テーブル
    ON  usegSUWMorph.TalkID = relSUWMorph2SUW.TalkID
   AND  usegSUWMorph.SUWMorphID = relSUWMorph2SUW. SUWMorphID
INNER JOIN  segSUW  Ⓑ
    ON  segSUW.TalkID = relSUWMorph2SUW.TalkID
   AND  segSUW.SUWID = relSUWMorph2SUW.SUWID
INNER JOIN  relSUW2Bunsetsu      ← Ⓑ と Ⓒ の親子関係テーブル
    ON  segSUW.TalkID = relSUW2Bunsetsu .TalkID
   AND  segSUW.SUWID = relSUW2Bunsetsu .SUWID
INNER JOIN  segBunsetsu  Ⓒ
    ON  segBunsetsu.TalkID = relSUW2Bunsetsu .TalkID
   AND  segBunsetsu.BunsetsuID = relSUW2Bunsetsu .BunsetsuID
 WHERE  relSUW2Bunsetsu.nth = relSUW2Bunsetsu.len      ← 文節最後の短単位
   AND  relSUWMorph2SUW.nth = relSUWMorph2SUW.len      ← 短単位最後の短単位形態素
   AND  usegSUWMorph.SUWPOS = "助詞"
```

CSJ-RDBの構成 ーリンク・テーブルー



リンク・テーブル　linkDepBunsetsu

文節同士の間の係り受け関係

■ linkDepBunsetsu テーブル

TalkID	BunsetsuID	ModifieeBunsetsuID
A01F0055	00395174L	00395851L

係り元の文節　　係り先の文節

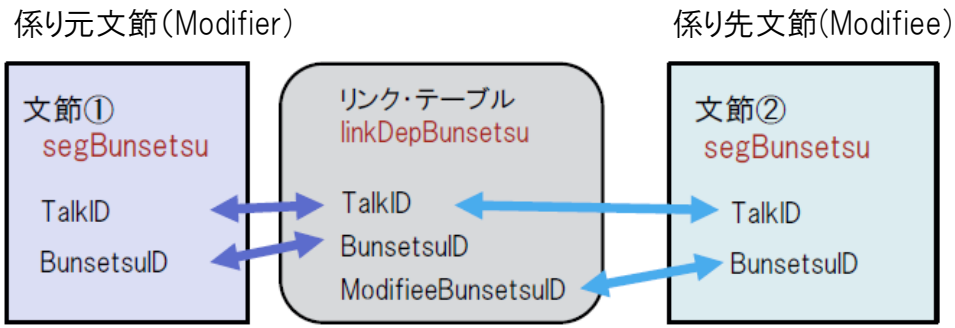
■ segBunsetsu テーブル

TalkID	BunsetsuID	StartTime	EndTime	Text
A01F0055	00395174L	395.173616	395.850548	こちらだけが
A01F0055	00395851L	395.850548	396.642817	点滅します

例題

例題20 linkDepBunsetsu を利用して係り元と係り先の文節テーブル（いずれもsegBunsetsu）を結合した上で、係り元の文節のテキストが「が」で終わるものを抽出し、TalkID、係り元と係り先の文節のテキストを選択

```
SELECT   Modifier.TalkID,  Modifier.Text,  Modifiee.Text
  FROM   segBunsetsu  AS  Modifier
INNER JOIN   linkDepBunsetsu
    ON   Modifier.BunsetsuID  =  linkDepBunsetsu.BunsetsuID
   AND   Modifier.TalkID  =  linkDepBunsetsu.TalkID
INNER JOIN   segBunsetsu  AS  Modifiee
    ON   Modifiee.TalkID  =  linkDepBunsetsu.TalkID
   AND   Modifiee.BunsetsuID  =  linkDepBunsetsu.ModifieeBunsetsuID
 WHERE   Modifier.Text  LIKE  "%が"
```



注意：文節①と文節②のテーブル名が同じなので問題が生じる。そのためテーブルに別名（例えば “Modifier”,“Modifiee”）を付ける。

リンク・テーブル　linkTone2AP

トーン情報とそのトーンが帰属するアクセント句との関係

■ pointTone テーブル

TalkID	ToneID	Time	toneLabel
A01F0055	28	11.77875	%L
A01F0055	29	11.89875	A
A01F0055	30	12.13375	L%
A01F0055	31	12.25875	H%

・・句頭が低いトーン(L)で開始
・・アクセント核(A)
・・句末の下降(L)
・・下降後に上昇(H)して句が終了

■ linkTone2AP テーブル

TalkID	ToneID	APID
A01F0055	28	00011679L
A01F0055	29	00011679L
A01F0055	30	00011679L
A01F0055	31	00011679L

帰属元のトーン　　帰属先のAP

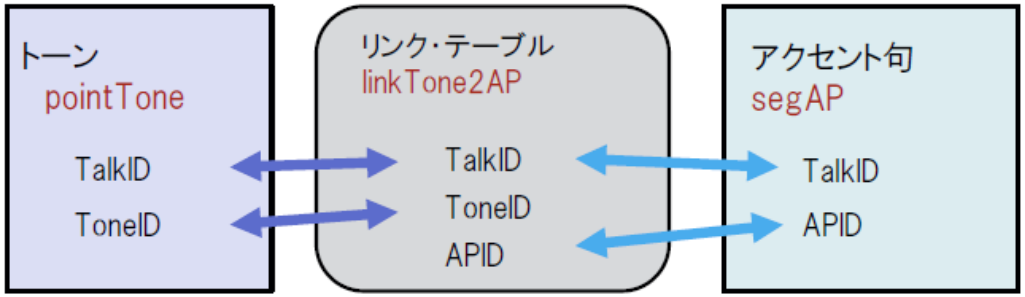
■ segAP テーブル

TalkID	APID	StartTime	EndTime	Text
A01F0055	00011679L	11.67941	12.362888	聴取に

例題

例題21　linkTone2AP を用いて帰属元のトーン(pointTone)と帰属先のアクセント句を結合した上で、トーン(toneLabel)がアクセント核“A”である行を抽出し、TalkID、APのテキストと開始・終了時刻、
トーン(この場合はアクセント核)の生起時間(Time)を選択

```
SELECT   segAP.TalkID,  segAP.Text,  segAP.StartTime,  segAP.EndTime, pointTone.Time
  FROM   pointTone
INNER JOIN   linkTone2AP
    ON   pointTone.TalkID  =  linkTone2AP.TalkID
   AND   pointTone.ToneID  =  linkTone2AP.ToneID
INNER JOIN   segAP
    ON   segAP.TalkID  =  linkTone2AP.TalkID
   AND   segAP.APID  =  linkTone2AP.APID
 WHERE   pointTone.toneLabel  =  "A"
```



例題

例題22 係り先がない(どこにも係らない)文節を抽出し、TalkID、文節ID、文節のテキストを表示

※ 文節テーブルに対して linkDepBunsetsu テーブルを外部結合し、
linkDepBunsetsu の列(例えばModifieeBunsetsuID)が NULL である行をWHERE句で選択。
値が NULL であるデータの検索には“IS NULL”演算子を用いる
(例： ModifieeBunsetsuID IS NULL)。

segBunsetsu

TalkID	BunsetsuID	Text
A01F0055	00007034L	(F えー)
A01F0055	00007208L	私共は
A01F0055	00008891L	乳児が
A01F0055	00009580L	音楽を
A01F0055	00009980L	どのように
A01F0055	00010420L	聞いているか
A01F0055	00011418L	また
A01F0055	00011679L	聴取に
A01F0055	00012363L	発達齢差が
A01F0055	00013335L	見られるかを
A01F0055	00014414L	検討しております

linkDepBunsetsu

TalkID	BunsetsuID	Modifiee BunsetsuID
A01F0055	00007208L	00014414L
A01F0055	00008891L	00010420L
A01F0055	00009580L	00010420L
A01F0055	00009980L	00010420L
A01F0055	00010420L	00013335L
A01F0055	00011679L	00013335L
A01F0055	00012363L	00013335L
A01F0055	00013335L	00014414L

外部結合後のテーブル

TalkID	BunsetsuID	Text	Modifiee BunsetsuID
A01F0055	00007034L	(F えー)	
A01F0055	00007208L	私共は	00014414L
A01F0055	00008891L	乳児が	00010420L
A01F0055	00009580L	音楽を	00010420L
A01F0055	00009980L	どのように	00010420L
A01F0055	00010420L	聞いているか	00013335L
A01F0055	00011418L	また	
A01F0055	00011679L	聴取に	00013335L
A01F0055	00012363L	発達齢差が	00013335L
A01F0055	00013335L	見られるかを	00014414L
A01F0055	00014414L	検討しております	

例題

例題22 係り先がない(どこにも係らない)文節を抽出し、TalkID、文節ID、文節のテキストを表示

```
SELECT  segBunsetsu.TalkID,  segBunsetsu.BunsetsuID,  segBunsetsu.Text
  FROM  segBunsetsu
LEFT  JOIN  linkDepBunsetsu
    ON  segBunsetsu.TalkID  =  linkDepBunsetsu.TalkID
   AND  segBunsetsu.BunsetsuID  =  linkDepBunsetsu.BunsetsuID
 WHERE  linkDepBunsetsu.ModifieeBunsetsuID  IS  NULL
```

← ModifieeBunsetsuID が NULL である行

第5部
SQLクエリ中級編

第5部　SQLクエリ中級編
① GROUP BY句

GROUP BY句

グループ毎の集計に用いる

1. 指定された列(たとえば品詞)に含まれるグループ毎(たとえば、名詞、動詞、形容詞、…)にデータをまとめ上げる
2. 集計関数を用いてグループ毎に集計

usegSUWMorph

TalkID	PlainOrthographicTranscription	SUWPOS	SUWConjugateForm
A01F0067	で	接続詞	
A01F0067	こちら	代名詞	
A01F0067	が	助詞	
A01F0067	時間	名詞	
A01F0067	情報	名詞	
A01F0067	のみ	助詞	
A01F0067	利用	名詞	
A01F0067	可能	形状詞	
A01F0067	な	助動詞	連体形
A01F0067	刺激	名詞	
A01F0067	で	助動詞	連用形
A01F0067	ある	動詞	終止形
A01F0067	と	助詞	
A01F0067	言え	動詞	連用形
A01F0067	ます	助動詞	終止形

SUWPOSで集計すると

SUWPOS	頻度
形状詞	1
助詞	3
助動詞	3
接続詞	1
代名詞	1
動詞	2
名詞	4

SUWPOSとSUWConjugateFormで集計すると

SUWPOS	SUWConjugateForm	頻度
形状詞		1
助詞		3
助動詞	終止形	1
助動詞	連体形	1
助動詞	連用形	1
接続詞		1
代名詞		1
動詞	終止形	1
動詞	連用形	1
名詞		4

GROUP BY句

行数(頻度)の取得

```
SELECT SUWPOS, COUNT(*) FROM usegSUWMorph GROUP BY SUWPOS
```

SUWPOS, : GROUP BY で指定した列

COUNT(*) : 集計関数

SUWPOS : まとめ上げの列

TalkID	PlainOrthographicTranscription	SUWPOS	SUWConjugateForm
A01F0067	で	接続詞	
A01F0067	こちら	代名詞	
A01F0067	が	助詞	
A01F0067	時間	名詞	
A01F0067	情報	名詞	
A01F0067	のみ	助詞	
A01F0067	利用	名詞	
A01F0067	可能	形状詞	
A01F0067	な	助動詞	連体形
A01F0067	刺激	名詞	
A01F0067	で	助動詞	連用形
A01F0067	ある	動詞	終止形
A01F0067	と	助詞	
A01F0067	言え	動詞	連用形
A01F0067	ます	助動詞	終止形

SUWPOS	COUNT(*)
形状詞	1
助詞	3
助動詞	3
接続詞	1
代名詞	1
動詞	2
名詞	4

GROUP BY句

行数(頻度)の取得

```
SELECT SUWPOS, SUWConjugateForm, COUNT(*) AS 頻度 FROM usegSUWMorph
GROUP BY SUWPOS, SUWConjugateForm
```

列名を「頻度」に

複数の列でまとめ上げ

TalkID	PlainOrthographicTranscription	SUWPOS	SUWConjugateForm
A01F0067	で	接続詞	
A01F0067	こちら	代名詞	
A01F0067	が	助詞	
A01F0067	時間	名詞	
A01F0067	情報	名詞	
A01F0067	のみ	助詞	
A01F0067	利用	名詞	
A01F0067	可能	形状詞	
A01F0067	な	助動詞	連体形
A01F0067	刺激	名詞	
A01F0067	で	助動詞	連用形
A01F0067	ある	動詞	終止形
A01F0067	と	助詞	
A01F0067	言え	動詞	連用形
A01F0067	ます	助動詞	終止形

SUWPOS	SUWConjugateForm	頻度
形状詞		1
助詞		3
助動詞	終止形	1
助動詞	連体形	1
助動詞	連用形	1
接続詞		1
代名詞		1
動詞	終止形	1
動詞	連用形	1
名詞		4

GROUP BY句　集計関数

■　GROUP BYとともに用いる主な集計関数

COUNT	行数を取得
SUM	合計値を取得
AVG	平均値を取得
MAX	最大値を取得
MIN	最小値を取得

例題

例題23 モーラテーブル(segMora)を対象に、モーラ記号(MoraEntity)毎に、
モーラ継続長の平均値を算出し、別名を“Avg”とした上で、
モーラ記号と合わせて表示

```
SELECT   segMora.MoraEntity,   Avg(EndTIme - StartTime)   AS   Avg
  FROM   segMora
GROUP BY segMora.MoraEntity
```

例題24 短単位形態論情報テーブルを対象に、品詞が「助詞」のものを抽出し、
短単位の代表表記と品詞細分類(SUWMiscPOSInfo1)毎に頻度を算出した上で、
別名を“Freq”とし、短単位の代表表記、品詞細分類と合わせて表示

```
SELECT  usegSUWMorph.SUWLemma,  usegSUWMorph.SUWMiscPOSInfo1,
        Count(*)  AS  Freq
  FROM  usegSUWMorph
 WHERE  usegSUWMorph.SUWPOS  =  "助詞"
GROUP BY  usegSUWMorph.SUWLemma,  usegSUWMorph.SUWMiscPOSInfo1
```

例題

例題25　短単位テーブル、短単位形態論情報テーブル、節単位テーブルを結合し、節単位の先頭の短単位を抽出した上で、短単位の品詞毎に頻度を算出し、別名を“Freq”として品詞と合わせて表示

```
SELECT  usegSUWMorph.SUWPOS,  Count(*)  AS  Freq
  FROM  usegSUWMorph
INNER JOIN  relSUWMorph2SUW
    ON  usegSUWMorph.TalkID  =  relSUWMorph2SUW.TalkID
   AND  usegSUWMorph.SUWMorphID  =  relSUWMorph2SUW.SUWMorphID
INNER JOIN  segSUW
    ON  segSUW.TalkID  =  relSUWMorph2SUW.TalkID
   AND  segSUW.SUWID  =  relSUWMorph2SUW.SUWID
INNER JOIN  relSUW2Clause
    ON  segSUW.TalkID  =  relSUW2Clause.TalkID
   AND  segSUW.SUWID  =  relSUW2Clause.SUWID
INNER JOIN  segClause
    ON  relSUW2Clause.TalkID  =  segClause.TalkID
   AND  relSUW2Clause.ClauseID  =  segClause.ClauseID
 WHERE  relSUW2Clause.nth  =  1
GROUP BY  usegSUWMorph.SUWPOS
```

第5部　SQLクエリ中級編
② ORDER BY句

ORDER BY句

- 昇順に並び替え

例）

```
ORDER BY SUWPOS
ORDER BY COUNT(*)          … GROUP BY した時
ORDER BY TALKID, SUWPOS… 複数の列で
```

- 降順に並び替え　下記のように列名の最後に DESC を追加

例）

```
ORDER BY SUWPOS DESC
ORDER BY TALKID, SUWPOS DESC
```

例題

例題26　例題25の結果を、頻度(“Freq”)の降順で並び替えて表示

```
SELECT  usegSUWMorph.SUWPOS,  Count(*)  AS  Freq
  FROM  usegSUWMorph
INNER JOIN  relSUWMorph2SUW
    ON  usegSUWMorph.TalkID  =  relSUWMorph2SUW.TalkID
   AND  usegSUWMorph.SUWMorphID  =  relSUWMorph2SUW.SUWMorphID
INNER JOIN  segSUW
    ON  segSUW.TalkID  =  relSUWMorph2SUW.TalkID
   AND  segSUW.SUWID  =  relSUWMorph2SUW.SUWID
INNER JOIN  relSUW2Clause
    ON  segSUW.TalkID  =  relSUW2Clause.TalkID
   AND  segSUW.SUWID  =  relSUW2Clause.SUWID
INNER JOIN  segClause
    ON  relSUW2Clause.TalkID  =  segClause.TalkID
   AND  relSUW2Clause.ClauseID  =  segClause.ClauseID
 WHERE  relSUW2Clause.nth  =  1
 GROUP BY  usegSUWMorph.SUWPOS
 ORDER BY  Freq  DESC
```

第5部　SQLクエリ中級編
③ HAVING句

HAVING句

- GROUP BY によるグループ毎のまとめ上げの後に、
  グループに対して条件で絞り込み（つまりGROUP BYとセット！）

グループで
まとめ上げ
（ GROUP BY ）

グループに対し
条件で絞り込み
（ HAVING）

※例えば件数が**1000以上**

TalkID	PlainOrthographic Transcription	SUWPOS
…	…	…
A01F0067	こちら	代名詞
A01F0067	が	助詞
A01F0067	時間	名詞
A01F0067	情報	名詞
A01F0067	のみ	助詞
A01F0067	利用	名詞
A01F0067	可能	形状詞
A01F0067	な	助動詞
A01F0067	刺激	名詞
A01F0067	で	助動詞
A01F0067	ある	動詞
A01F0067	と	助詞
A01F0067	言え	動詞
A01F0067	ます	助動詞
…	…	…

SUWPOS	頻度
代名詞	809
副詞	1516
助動詞	4381
助詞	10686
動詞	4298
名詞	6812
形容詞	493
形状詞	571
感動詞	3435
接尾辞	819
接続詞	405
接頭辞	173
言いよどみ	608
記号	28
連体詞	362

SUWPOS	頻度
副詞	1516
助動詞	4381
助詞	10686
動詞	4298
名詞	6812
感動詞	3435

※SQLで書くと

```
SELECT   SUWPOS,
         COUNT(*)  AS Freq
  FROM   usegSUWMorph
GROUP BY SUWPOS
 HAVING  Freq  >=  1000
```

例題

例題27 短単位形態論情報テーブルを対象に、短単位の代表表記と品詞毎に頻度を算出し、別名を“Freq”とした上で、頻度が500以上のものに絞り込み、短単位の代表表記、品詞、頻度を、頻度の降順で表示

```
SELECT   usegSUWMorph.SUWLemma,   usegSUWMorph.SUWPOS,
         Count(*)  AS  Freq
  FROM   usegSUWMorph
GROUP BY  usegSUWMorph.SUWLemma, usegSUWMorph.SUWPOS
  HAVING  Freq  >=  500
ORDER BY  Freq   DESC
```

例題

例題28 短単位形態論情報テーブルを対象に、品詞細分類が「格助詞」のものを抽出し、
短単位の代表表記毎に頻度を算出した上で、別名を ”Freq” とし、
頻度が500以上のものに絞り込んだ上で、
短単位の代表表記と頻度を、頻度の降順で表示

```
  SELECT   usegSUWMorph.SUWLemma,  Count(*)  AS  Freq
    FROM   usegSUWMorph
   WHERE   usegSUWMorph.SUWMiscPOSInfo1  =  ″格助詞″
GROUP BY   usegSUWMorph.SUWLemma
  HAVING   Freq >= 500
ORDER BY   Freq   DESC
```

SQLの構文　まとめ

必須	SELECT	列名1，列名2，…	SELECT文
必須	FROM	テーブル名	
任意	INNER JOIN	テーブル名2	JOIN句
任意	WHERE	抽出条件	WHERE句
任意	GROUP BY	グループ化する列名	GROUP BY句
任意	HAVING	グループに対する抽出条件	HAVING句
任意	ORDER BY	整列対象とする列名	ORDER BY句

文と句は必ずこの順番！

※　JOIN句にはLEFT OUTER JOINもあり。
　　またJOIN句は複数回繰り返すことも可。

第6部
SQLクエリ中上級編

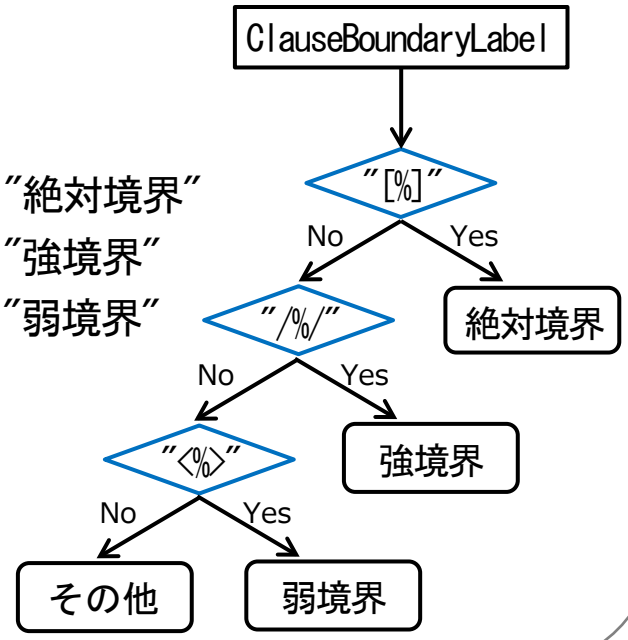
第6部　SQLクエリ中上級編
① CASE式

CASE式

- 条件式によって異なる値を返す

例） SELECT 文でのCASE式の利用

```
SELECT TalkID, ClauseID, ClauseBoundaryLabel,
       CASE
         WHEN ClauseBoundaryLabel LIKE "[%]" THEN "絶対境界"
         WHEN ClauseBoundaryLabel LIKE "/%/" THEN "強境界"
         WHEN ClauseBoundaryLabel LIKE "<%>" THEN "弱境界"
         ELSE "その他"
       END AS ClauseType ← 列に別名を付与
FROM segClause
```



※ ClauseBoundaryLabel の例：“[文末]” “/並列節ガ/” “<理由節ノデ>”

CASE式を利用した集計

- 条件式によって得られる値毎にまとめ上げ

例） SELECT 文と GROUP BY句でのCASE式の利用

```
SELECT CASE
          WHEN  ClauseBoundaryLabel  LIKE  "[%]" THEN  "絶対境界"
          WHEN  ClauseBoundaryLabel  LIKE  "/%/" THEN  "強境界"
          WHEN  ClauseBoundaryLabel  LIKE  "<%>" THEN  "弱境界"
          ELSE  "その他"
       END AS  ClauseType,
       COUNT(*) AS Freq← 列に別名を付与
FROM  segClause
GROUP BY ClauseType
                        ← 条件式（の別名）でグループ化することもできる
```

例題

例題29 短単位形態論情報テーブルを対象に、短単位を「活用語」と「非活用語」に分け、SUWType とした上で、SUWType毎の頻度を算出し、別名を “Freq” としてSUWType と合わせて表示

```
SELECT   CASE
            WHEN   SUWConjugateForm = ""   THEN   "非活用語"
            ELSE   "活用語"
         END   AS  SUWType,
         Count(*)  AS  Freq
   FROM   usegSUWMorph
GROUP BY  SUWType
```

※ 非活用語は活用形が空(SUWConjugateForm = “”)

例題

例題30 モーラテーブルを対象に、モーラ記号を「特殊拍」（「ン」「ッ」「ー」）と
それ以外の「一般拍」に分け、MoraType とした上で、MoraType 毎の継続長の
平均値を算出し、別名を“Duration”としてMoraTypeと合わせて表示

```
SELECT  CASE
            WHEN segMora.MoraEntity  IN  (″ン″,″ッ″,″ー″)  THEN  ″特殊拍″
            ELSE ″一般拍″
        END  AS  MoraType,
        Avg(EndTime - StartTime )  AS  Duration
  FROM  segMora
GROUP BY  MoraType
```

※ 「x = “a” OR x = “b”」 は「x IN (“a”, ”b”)」 と簡潔に書くことができる

例題

例題31　ファイル毎に、全単語(長単位)に占める「言いよどみ」の割合を算出し、rateDisf という別名を付けた上で、ファイル名と合わせて表示

```
SELECT  TalkID,
        SUM(CASE WHEN  LUWPOS = "言いよどみ"
              THEN  1.0  END) /
        COUNT( * )  AS  rateDisf
   FROM  usegLUWMorph
GROUP BY  TalkID
```

名詞の場合に1.0とし、ファイル毎にその合計(言いよどみの頻度)を算出。1ではなく1.0とするのは、そのあと割り算した時に結果を小数で求めるための工夫(整数同士の四則演算では整数値しか返さない)。

全単語(名詞などの条件なし)の合計をファイル毎に求め、その値で名詞頻度を割り、名詞率を算出

第6部　SQLクエリ中上級編
② 副クエリー

副クエリー

以下の例のように、主たるクエリー文の中に、副クエリーを埋め込むことができる

```
SELECT *, EndTime - StartTime AS duration FROM segMora
 WHERE EndTime - StartTime >
        ( SELECT AVG (EndTime - StartTime ) FROM segMora )
```

↓ 副クエリー

モーラの平均継続時間長を計算

※ 主クエリーのWHERE句では、モーラの継続時間長(EndTime - StartTime)が、「モーラの平均継続時間長」よりも長いもの、という条件を設定。ここでは「モーラの平均継続時間長」を求めるために、副クエリーを用いている。

副クエリー

副クエリーは、SELECT文やJOIN句などにも含めることができる

```
SELECT  mora.TalkID, mora. avgMora,  SUW.avgSUW,  AP.avgAP
    FROM (SELECT  TalkID,  AVG(EndTime - StartTime)  AS  avgMora
          FROM  segMora  GROUP  BY  TalkID  )  AS  mora
INNER JOIN (SELECT  TalkID,  AVG(EndTime - StartTime)  AS  avgSUW
          FROM  segSUW  GROUP  BY  TalkID  )  AS  SUW
    ON  mora.TalkID  =  SUW.TalkID
INNER JOIN (SELECT  TalkID,  AVG(EndTime - StartTime)  AS  avgAP
          FROM  segAP  GROUP  BY  TalkID  )  AS  AP
    ON  mora.TalkID  =  AP.TalkID
```

ファイルごとにモーラの平均継続時間長を計算
結果に別名 mora を付与

ファイルごとに短単位の平均継続時間長を計算
結果に別名 SUW を付与

TalkIDをキーにmora（上記副クエリーの結果）と結合

ファイルごとにAPの平均継続時間長を計算
結果に別名 AP を付与

TalkIDをキーにmora（上記副クエリーの結果）と結合

第6部　SQLクエリ中上級編
③ テーブルの作成

テーブルの作成

クエリーで得られた結果に名前を付け、新しいテーブルを作成

```
CREATE TABLE mora AS
SELECT TalkID, AVG(EndTime - StartTime) AS avgMora
FROM segMora GROUP BY TalkID ;
```

SELECT文の結果を mora という名でテーブルとして保存

; は複数の文をまとめて書く場合に文末に記す

```
CREATE TABLE SUW AS
SELECT TalkID, AVG(EndTime - StartTime) AS avgSUW
FROM segSUW GROUP BY TalkID ;
```

SELECT文の結果を SUW という名でテーブルとして保存

```
CREATE TABLE AP AS
SELECT TalkID, AVG(EndTime - StartTime) AS avgAP
FROM segAP GROUP BY TalkID ;
```

SELECT文の結果を AP という名でテーブルとして保存

```
SELECT mora.TalkID, mora. avgMora, SUW.avgSUW, AP.avgAP
FROM mora
INNER JOIN SUW ON mora.TalkID = SUW.TalkID
INNER JOIN AP ON mora.TalkID = AP.TalkID ;
```

新規テーブルを用いたSELECT文
2頁前の副クエリーの例と比較

実際にテーブルを作成してしまうため、よほど頻繁に使うものだけに留め、
副クエリーや後述の一時テーブルの作成、ビューなどを用いたほうがよい

一時テーブルの作成

クエリーで得られた結果に名前を付け、新しい一時テーブル(TEMPORARY TABLE)を作成

```
CREATE TEMPORARY TABLE mora AS
SELECT TalkID, AVG(EndTime - StartTime) AS avgMora
FROM segMora GROUP BY TalkID ;
```

SELECT文の結果を mora という名で一時テーブルとして保存

```
CREATE TEMPORARY TABLE SUW AS
SELECT TalkID, AVG(EndTime - StartTime) AS avgSUW
FROM segSUW GROUP BY TalkID ;
```

SELECT文の結果を SUW という名で一時テーブルとして保存

```
CREATE TEMPORARY TABLE AP AS
SELECT TalkID, AVG(EndTime - StartTime) AS avgAP
FROM segAP GROUP BY TalkID ;
```

SELECT文の結果を AP という名で一時テーブルとして保存

```
SELECT mora.TalkID, mora. avgMora, SUW.avgSUW, AP.avgAP
FROM mora
INNER JOIN SUW ON mora.TalkID = SUW.TalkID
INNER JOIN AP ON mora.TalkID = AP.TalkID
```

新規一時テーブルを用いたSELECT文

副クエリーを用いて1つのSELECT文で表現すると複雑になる場合などに用いる(3頁前の例と比較)
セッション終了後に一時テーブルは削除される。何度も利用するものは後述のビューが便利

第6部　SQLクエリ中上級編
④ ビュー（仮想の表）

ビュー（仮想の表）

以下の例のように、クエリーで得られた結果に名前を付け、再利用できるようにした仮想の表

```
CREATE VIEW mora AS
SELECT TalkID, AVG(EndTime - StartTime) AS avgMora
FROM segMora GROUP BY TalkID ;
```

SELECT文の結果を mora という名でビューとして保存

```
CREATE VIEW SUW AS
SELECT TalkID, AVG(EndTime - StartTime) AS avgSUW
FROM segSUW GROUP BY TalkID ;
```

SELECT文の結果を SUW という名でビューとして保存

```
CREATE VIEW AP AS
SELECT TalkID, AVG(EndTime - StartTime) AS avgAP
FROM segAP GROUP BY TalkID ;
```

SELECT文の結果を AP という名でビューとして保存

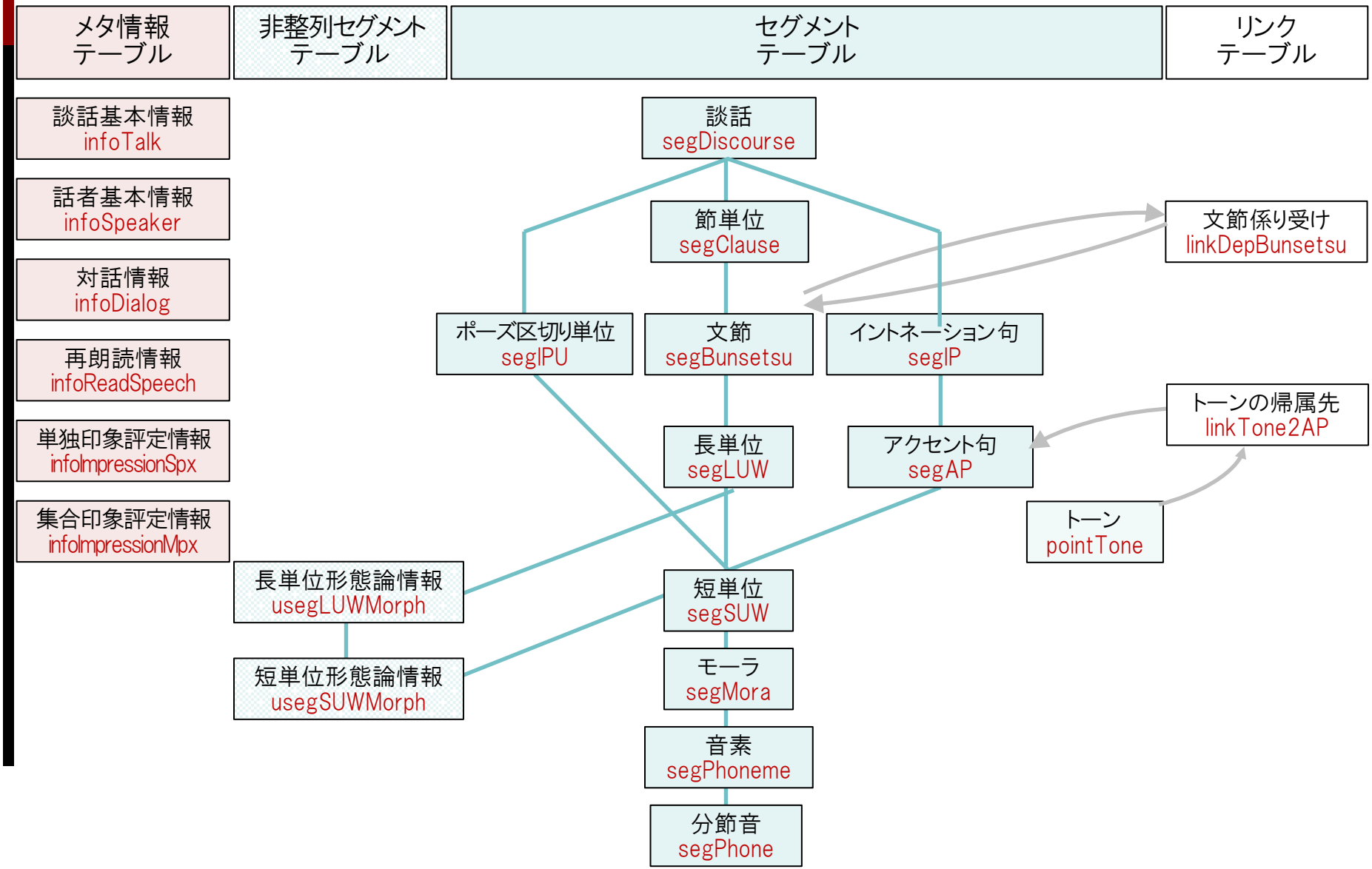
```
SELECT mora.TalkID, mora. avgMora, SUW.avgSUW, AP.avgAP
FROM mora
INNER JOIN SUW ON mora.TalkID = SUW.TalkID
INNER JOIN AP ON mora.TalkID = AP.TalkID ;
```

保存したビューmoraの利用
保存したビューSUWの利用
保存したビューAPの利用

ビューは実際のテーブルが作成されるわけではなく、実行するたび計算される
セッションが終了してもビューは残るため、別セッションでも再利用したい場合に利用するとよい

第7部
メタ情報テーブル

メタ情報テーブル


メタ情報
テーブル
非整列セグメント
テーブル
セグメント
テーブル
リンク
テーブル
談話基本情報
infoTalk
話者基本情報
infoSpeaker
対話情報
infoDialog
再朗読情報
infoReadSpeech
単独印象評定情報
infoImpressionSpx
集合印象評定情報
infoImpressionMpx
談話
segDiscourse
節単位
segClause
文節係り受け
linkDepBunsetsu
ポーズ区切り単位
segIPU
文節
segBunsetsu
イントネーション句
segIP
トーンの帰属先
linkTone2AP
長単位
segLUW
アクセント句
segAP
トーン
pointTone
長単位形態論情報
usegLUWMorph
短単位
segSUW
短単位形態論情報
usegSUWMorph
モーラ
segMora
音素
segPhoneme
分節音
segPhone

談話基本情報テーブル

列名	説明	例
TalkID	談話ID	S01F0001
TalkType	談話タイプ	monolog
Genre	ジャンル	学会/模擬
SpeakerID	話者ID	116
NumAudience	聴き手人数	50
Topic	学会種・模擬講演のテーマ等	工学系学会1(音声関係)
Style	学会における講演形式	通常の発表
Device	講演使用器材・配布資料	OHP・スライドの類
SpeechPreparation	講演の準備	原稿
SpeechSkill	講演の得手・不得手	得意
TalkExperience	講演経験	10(年)
EducationBackground	最終学歴	学部卒
SpeakerAge	収録時の年齢	35to39

TalkIDをキーに、これまで扱ってきた単位テーブルなどと結合して利用

例題

例題32　短単位形態論情報と談話基本情報を結合し、短単位の品詞と談話タイプごとの頻度を算出

```
SELECT   usegSUWMorph.SUWPOS,  infoTalk.TalkType,  COUNT(*)  AS  頻度
  FROM   usegSUWMorph
INNER JOIN  infoTalk
    ON   usegSUWMorph.TalkID  =  infoTalk.TalkID
GROUP BY  usegSUWMorph.SUWPOS,   infoTalk.TalkType
```

話者基本情報テーブル

列名	説明	例
SpeakerID	話者ID	116
SpeakerSex	話者性別	男/女
SpeakerBirthGeneration	話者生年代(5年刻み)	70to74(1970-74年生まれ)
SpeakerBirthPlace	話者出生地	東京都
ResidenceLength	居住年数	首都圏:55年
ResidenceLengthCriticalPeriod	居住年数(言語形成期)	首都圏:12年
FatherBirthPlace	父出身地	秋田県
MotherBirthPlace	母出身地	新潟県

話者基本情報テーブルにはTalkIDが存在しないため、
まず SpeakerIDをキーに談話基本情報テーブルと結合した上で、
談話基本情報テーブルのTalkIDをキーに、単位テーブルなどと結合する
(4頁目の「RDBとは」の例を参照)

例題

例題33　話者基本情報、談話基本情報、短単位形態論情報を結合し、
短単位の品詞と性別ごとの頻度を算出

```
    SELECT  usegSUWMorph.SUWPOS,  infoSpeaker.SpeakerSex,  COUNT(*)  AS  頻度
      FROM  infoSpeaker
INNER JOIN  infoTalk
        ON  infoSpeaker.SpeakerID  =  infoTalk.SpeakerID
INNER JOIN  usegSUWMorph
        ON  infoTalk.TalkID  =  usegSUWMorph.TalkID
 GROUP BY  usegSUWMorph.SUWPOS,  infoSpeaker. SpeakerSex
```

集合印象評定情報テーブル

列名	説明	例
ImpressionID	印象評定ID	5
TalkID	談話ID	S01F0001
Position	評定位置	2
Rater	評定者	2f1
A01	好きな－嫌いな	7(好きな)－1(嫌いな)
A02	心地よい－不快な	7(心地よい)－1(不快な)
A03	感じの良い－感じの悪い	7(感じの良い)－1(感じの悪い)
A04	親しみやすい－親しみにくい	7(親しみやすい)－1(親しみにくい)
A05	流暢な－たどたどしい	7(流暢な)－1(たどたどしい)
(省略)		
A07	なめらかな－しどろもどろな	7(なめらかな)－1(しどろもどろな)
A08	上手い－下手な	7(上手い)－1(下手な)
A09	速い－遅い	7(速い)－1(遅い)
A10	スピード感のある－ゆったりした	7(スピード感のある)－1(ゆったりした)
A11	せわしげな－のんきな	7(せわしげな)－1(のんきな)
A12	落ち着きのない－落ち着きのある	7(落ち着きのない)－1(落ち着きのある)
A13	声の大きい－声の小さい	7(声の大きい)－1(声の小さい)
(省略)		
A21	あらたまった－くだけた	7(あらたまった)－1(くだけた)
A22	きまじめな－奔放な	7(きまじめな)－1(奔放な)
A23	きちんとした－くつろいだ	7(きちんとした)－1(くつろいだ)
A24	甘えた－そっけない	7(甘えた)－1(そっけない)
A25	その場で考えて話している－原稿を読み上げている	7(その場で考えて話している)－1(原稿を読み上げている)
A26	聞き取りやすい－聞き取りにくい	7(聞き取りやすい)－1(聞き取りにくい)
(省略)		

TalkIDをキーに、これまで扱ってきた単位テーブルなどと結合して利用

1つのファイル(TalkID)に対し、10人の評定者が3回(合計30回)評定している点に注意

例題

例題34　複合印象評定情報を対象に、ファイル毎にA05（流暢性）とA21（あらたまり度）の30回分の評定値の平均を算出し、評定値平均が2以下を”L”、5以上を”H”、3～5を”M”として、結果を一時テーブル rating に保存

```
CREATE TEMPORARY TABLE rating AS
 SELECT  TalkID,
         CASE
            WHEN  AVG(A05) <= 3 THEN  'L'
            WHEN  AVG(A05) >= 5 THEN  'H'
            ELSE  'M'
          END  AS  fluency,
          CASE
            WHEN  AVG(A21) <= 3  THEN  'L'
            WHEN  AVG(A21) >= 5  THEN  'H'
            ELSE  'M'
          END  AS  formality
   FROM  infoImpressionMpx
GROUP BY  TalkID
```

例題

例題35 例題31(ファイル毎に、全単語(長単位)に占める「言いよどみ」の割合を算出し rateDisf という別名を付けた上で、ファイル名と合わせて表示)を一時ファイル disf として作成した上で(回答省略)、例題34で作成した一時ファイル rating と TalkID をキーに結合し、流暢性(fluency、H,M,L 3段階)ごとに名詞率の平均を算出

```
SELECT  rating.fluency,  AVG(disf.rateDisf)  AS  rateDisf
  FROM  rating
INNER JOIN  disf
        ON  rating.TalkID  =  disf.TalkID
 GROUP BY  rating.fluency
```