SQL

Structured Query Language

§3.1 SQLの概要

- 最も重要なリレーショナル・データ操作言語
- 標準(ANSI, ISO, JIS)
- 多くの商用のDBMS製品で利用可能
- IBMのSEQUELから発展
  - 1970, IBM San Jose研究施設

SQLの特徴

- 対話的な使用
- 応用プログラムへの埋め込み
  - データ・サブ言語, データ・アクセス言語
- 入力と出力がともにリレーション
- リレーショナル完備
  - DB言語の能力を表す基準.
  - リレーショナル代数の記述力を持つDB言語
- リレーショナル・モデルとの相違点
  - 重複したタプルの存在.
  - 属性やタプルに順序づけがされている.

§3.2 単一テーブルの問い合わせ

- 射影, 選択, ソート,
- 組み込み関数,
- グループ化
- 例で扱うリレーション

student

sid	name	major	gradelevel	age
100	Jones	History	GR	21
150	Parks	Accounting	SO	19
200	Baker	Math	GR	50
250	Glass	History	SN	50
300	Baker	Accounting	SN	41
350	Russell	Math	JR	20
400	Rye	Accounting	FR	18
450	Jones	History	SN	24

射影

SELECT sid, name, major
FROM student

sid	name	major
100	Jones	History
150	Parks	Accounting
200	Baker	Math
250	Glass	History
300	Baker	Accounting
350	Russell	Math
400	Rye	Accounting
450	Jones	History

重複タプルを含む結果の例

SELECT major FROM student

major
History
Accounting
Math
History
Accounting
Math
Accounting
History

重複タプルを削除する(DISTINCT)

SELECT DISTINCT major FROM student

major
Accounting
History
Math

選択(WHERE)

```
SELECT sid, name, major, gradelevel, age
FROM student
WHERE major='Math'
```

sid	name	major	gradelevel	age
200	Baker	Math	GR	50
350	Russell	Math	JR	20

全属性の指定には*マークを用いる.

```
SELECT * FROM student WHERE major='Math'
```

射影と選択の組み合わせ

SELECT name, age FROM student
WHERE major='Math'

name	age
Baker	50
Russell	20

複数の条件指定

SELECT name, age FROM student
WHERE major='Math' AND age > 21

name	age
Baker	50

集合による指定(IN)

SELECT name, major FROM student
WHERE major IN ['Math','Accounting']

name	major
Parks	Accounting
Baker	Math
Baker	Accounting
Russell	Math
Rye	Accounting

標準の[ ] の代わりに ( )を使うシステムがある.

集合による指定(NOT IN)

SELECT name, major FROM student
WHERE major NOT IN ['Math','Accounting']

name	major
Jones	History
Glass	History
Jones	History

指定したどれとも異なる属性値が選ばれる.

タプルのソート(ORDER BY)

```
SELECT name, major, age
FROM   student
WHERE  major = 'Accounting'
ORDER BY name
```

name	major	age
Baker	Accounting	41
Parks	Accounting	19
Rye	Accounting	18

ソート：複数列, 降順/昇順の指定

```
SELECT name, major, age
FROM  student
WHERE gradelevel IN ['FR','SO','SN']
ORDER BY major ASC, age DESC
```

name	major	age
Baker	Accounting	41
Parks	Accounting	19
Rye	Accounting	18
Glass	History	50
Jones	History	24

SQLの組み込み関数

集約(Aggregates)関数

- COUNT： タプルの数
- SUM:　数値属性の合計
- AVG:　 平均値
- MAX:　 最大値
- MIN:　 最小値

COUNTの例 1 :学生の数を知るには?

SELECT COUNT(*) FROM　student

count
8

組み込み関数は基本的に属性指定と同時に使用できない.

×SELECT name, count(*) FROM student

ただし, GROUP BYは例外. 後で説明.

COUNTの例２, majorの数を知るには?

SELECT COUNT(major) FROM student

count
8

SELECT COUNT(DISTINCT major)
FROM student

count
3

COUNTの例3

以下の二つは同じであるが, 後者を使用できないシステムが多い.(関数をwhereの一部に使うことができない)

```
SELECT MAX(age) FROM student
```

```
SELECT age FROM student
WHERE age = max(age)
```

max
50

組み込み関数とグループ化

SELECT major, COUNT(*) FROM student
GROUP BY major

major	count
Accounting	3
History	3
Math	2

グループ化と絞り込み

- グループ化した表に制限をつけにはHAVINGを使う.
- HAVINGは, 通常, GROUP BY とともに指定される.

SELECT major, COUNT(*) FROM student

GROUP BY major HAVING COUNT(*) > 2

major	count
Accounting	3
History	3

HAVINGとWHEREの適用順序

SQL標準では WHEREが先.

```
SELECT major, AVG(age) FROM student
WHERE  gradelevel='SN'
GROUP  BY major
HAVING COUNT(*) >= 2
```

major	avg
History	37

LIKE述語

属性値が文字列の場合に, 一部を指定して検索できる

```
SELECT * FROM student
WHERE  major LIKE ’A%’
```

```
SELECT * FROM student
WHERE gradelevel LIKE ’S_’
```

- %は0以上の任意の n 文字からなる文字列
- _は任意の 1 文字