講義補助資料 No.3 (10/16)

67

SQL

- 最も標準的なリレーショナルデータベースの言語
- ISOによる**国際標準規格**であり特定の企業に依存しない
  - SQLの規格：SQL89(SQL1), SQL92(SQL2), SQL:1999(SQL3), SQL:2003, SQL:2006, SQL:2008, SQL:2011, **SQL:2016**
  - 標準規格としてのSQLは「何かの略語ではない」と規定されている（参考：IBM社の製品で使われているSQLはStructured Query Languageの略とされている）
- 実際のリレーショナルデータベース管理システムでは、**SQL:2016**に準拠した実装が多い

68

SQLの特徴

- SQLは表形式データの操作言語(manipulation language)であり、複数の表を入力として、1つの表を作成して返す
- ここでいう「表(table)」とはタップルのマルチ集合（同じデータが複数存在することを許す）である
  - タップルは表の「行」のデータを指す
- SQLでは他の多くのプログラミング言語とは異なり、データ操作を英語に似た構文で宣言的に記述する
  - 宣言的な記述は、処理系に最適化のための自由度を与える
  - 手続き的な記述では、データ操作の順番を変えるなどの最適化が容易ではない

69

SQLの構文（一部）

select [all | distinct] <列リスト> from <表リスト>
[where <検索条件>]
[group by <グループ化する列のリスト>]
[having <タップルの選択条件>]
[union [all] | intersect | except] 副問合せ]
[order by <ソート条件> [asc | desc]]

（注）[ ]の中は省略可能であり、[a | b]はaかbを選択するか、または省略できることを示す

70

SQLの構文の意味

select 列リスト from 表リスト where 検索条件

- 実際の処理手順は from-where-selectの順で考えたほうが理解しやすい

例　select R.A from R,S where R.A=S.B

from R, S ・・・・・・・・・R×S（RとSの直積）

where R.A=S.B・・・・R.A=S.Bの条件で選択

select R.A ・・・・・・・・R.Aのみを出力

（注意）select **distinct** R.Aとしない限り、行の重複は除かれない

71

SQLの検索条件

select [all | distinct]<列リスト> from <表リスト> [where **<検索条件>**]

- <検索条件>では，次の比較演算子を利用することができる
  =, <>, >, <, <=, >=, like
- likeによる部分文字列比較の使用例：
  比較対象が “employee”のとき、
  like “emp%”　“emp”で始まる文字列、
  like “%loyee”　“loyee”で終わる文字列、
  like “%loy%”　“loy”を含む文字列
  は、いずれも真となる
- 比較演算子の式は、andやorでつなぐことができる
  （例） where salary >= 50 and salary <= 100
- 他にin述語やexists述語を書くことができる（「部分問合せ」で説明）

72

SQLによる問合せの種類

- 単純問合せ(simple query)
  - 1つの表(fromの後に1個しか表を書かない)についての問合せ
- 結合問合せ(join query)
  - 複数の表をfromの後に書く問合せ
  - 表を結合するために表の直積を行った後、検索条件の下で選択し(whereの条件)、selectの後の列リストで必要な部分だけ取り出して出力
- 部分問合せ（入れ子型問合せ、nested query）
  - whereの検索条件の指定の中に、別のSQLの問合せを埋め込む
  - 結合問合せとは異なり、結合処理は入れ子の最も内部の問合せから処理される

講義補助資料 No.3 (10/16)

73

問合せの例:使用する表

emp

empno	ename	deptno	salary	roomno
E01	Smith	D01	100	R01
E02	Morgan	D01	70	R03
E03	Robert	D01	80	R02
E04	Washington	D02	120	R10
E05	Lincoln	D02	90	R11
E06	Taylor	D01	80	R12
E91	Suzuki	D91	140	R91
E92	Tanaka	D91	130	R91
E93	Matsuda	D91	90	R91

dept

deptno	dname	manager
D01	Account	E01
D02	Personnel	E04
D91	Database	E91

74

問合せの例の意味

- emp(社員表)
  - ➢empno(社員番号)
  - ➢ename(社員名)
  - ➢deptno(部門番号)
  - ➢salary(給与)
  - ➢roomno(部屋番号)
- dept(部門表)
  - ➢deptno(部門番号)
  - ➢dname(部門名)
  - ➢manager(部門長)

75

単純問合せの例(1)

select * from emp　（表empを出力）

SQLの問合せは必ずselectで始まる

empno	ename	deptno	salary	roomno
E01	Smith	D01	100	R01
E02	Morgan	D01	70	R03
E03	Robert	D01	80	R02
E04	Washington	D02	120	R10
E05	Lincoln	D02	90	R11
E06	Taylor	D01	80	R12
E91	Suzuki	D91	140	R91
E92	Tanaka	D91	130	R91
E93	Matsuda	D91	90	R91

76

単純問合せの例(2)

select [all] deptno from emp

（表empのdeptnoの一覧を求めよ）

deptno
D01
D01
D01
D02
D02
D01
D91
D91
D91

select **distinct** deptno from emp

（重複を削除）

deptno
D01
D02
D91

77

単純問合せの例(3)

select * from emp
where salary >= 100
（給与が100以上の社員を求めよ）

empno	ename	deptno	salary	roomno
E01	Smith	D01	100	R01
E04	Washington	D02	120	R10
E91	Suzuki	D91	140	R91
E92	Tanaka	D91	130	R91

78

集約演算とグループ化

集約演算(aggregation operation)

問合せの結果得られる列に対して行う演算のこと

- 演算の種類は、avg（平均）、sum（総和）、max（最大）、min（最小値）、count（要素の個数）など
- group byを使ったグループ化(grouping)により、特定の項目ごとの集約演算が行える
  例:部門ごとの給与の平均，授業科目ごとの成績の平均
- グループ化では、さらにhaving条件によりグループ化する列の項目に条件を付けることができる
  例:**3人以上社員がいる**部門での給与の平均値、
  5人以上受講者がいる授業科目での成績の平均値

79

集約演算の例(1)

グループ化しない集約演算
select **avg**(salary) from emp where deptno = 'D01'
（表empからdeptnoがD01のsalaryの平均値を求めよ）

avg(salary)
82.5

グループ化した集約演算
select deptno, **avg**(salary) from emp **group by** deptno
（表empからdeptnoごとのsalaryの平均値を求めよ）

deptno	avg(salary)
D01	82.5
D02	105
D91	120

80

集約演算の例(2)

having条件付きでグループ化した集約演算
select deptno, **avg**(salary) from emp **group by** deptno
having count(*) >= 3
（表empからタップルが3個以上あるdeptnoについて、そのdeptnoのsalaryの平均値を求めよ）

deptno	avg(salary)
D01	82.5
D91	120

81

結合問合せ(join query)

複数の表を指定して、それらの表にまたがって検索条件を満たすタップルを求める問合せのこと

- 意味的には、指定された複数の表のタップルの直積を取った後、検索条件を満たすものを選択する操作
- 問合せが複数の表にまたがるため、列名を指定するときは表名を前につける必要がある
  - 例：表empの中の列enameであれば、**emp.**enameと書く
- 同じ表を2回以上使って、表にまたがる結合問合せを行うときは、**表名では区別が付かない**ので、表に**別名(alias)**を付けて区別することができる

82

結合問合せの例

select emp.*, dept.* from emp, dept
where emp.deptno=dept.deptno
（注）SQL:1999から以下の構文が導入された
select emp.*, dept.* from emp **join** dept
on emp.deptno=dept.deptno

empno	ename	deptno	salary	roomno	deptno	dname	manager
E01	Smith	D01	100	R01	D01	Account	E01
E02	Morgan	D01	70	R03	D01	Account	E01
E03	Robert	D01	80	R02	D01	Account	E01
E04	Washington	D02	120	R10	D02	Personnel	E04
E05	Lincoln	D02	90	R11	D02	Personnel	E04
E06	Taylor	D01	80	R12	D01	Account	E01
E91	Suzuki	D91	140	R91	D91	Database	E91
E92	Tanaka	D91	130	R91	D91	Database	E91
E93	Matsuda	D91	90	R91	D91	Database	E91

83

結合問合せの例（自然結合）

select * from emp **natural join** dept

SQL:1999で導入された構文
列の順番が必ずしも元の表と同じ順番になるとは限らないことに注意（結合する2個の表に共通する列名（以下ではdeptno)が先頭に来る

deptno	empno	ename	salary	roomno	dname	manager
D01	E01	Smith	100	R01	Account	E01
D01	E02	Morgan	70	R03	Account	E01
D01	E03	Robert	80	R02	Account	E01
D02	E04	Washington	120	R10	Personnel	E04
D02	E05	Lincoln	90	R11	Personnel	E04
D01	E06	Taylor	80	R12	Account	E01
D91	E91	Suzuki	140	R91	Database	E91
D91	E92	Tanaka	130	R91	Database	E91
D91	E93	Matsuda	90	R91	Database	E91

84

結合問合せの例（自然結合の別の書き方）

select emp.*, dept.dname, dept.manager
from emp, dept where emp.deptno=dept.deptno
または
select emp.*, dept.dname, dept.manager
from emp **join** dept **on** emp.deptno=dept.deptno

empno	ename	deptno	salary	roomno	dname	manager
E01	Smith	D01	100	R01	Account	E01
E02	Morgan	D01	70	R03	Account	E01
E03	Robert	D01	80	R02	Account	E01
E04	Washington	D02	120	R10	Personnel	E04
E05	Lincoln	D02	90	R11	Personnel	E04
E06	Taylor	D01	80	R12	Account	E01
E91	Suzuki	D91	140	R91	Database	E91
E92	Tanaka	D91	130	R91	Database	E91
E93	Matsuda	D91	90	R91	Database	E91

85

結合問合せでの別名の使用例

- 表名を使った結合問合せ（SQL:1999構文）

```
select emp.ename from emp join dept
   on emp.deptno = dept.deptno
   where  dept.dname = 'Account'
```

（表empと表deptから、deptnoのdnameがAccountである者のename を求めよ）

- **別名**を使った結合問合せ

```
select a.ename from emp a join dept b
   on a.deptno = b.deptno
   where b.dname = 'Account'
```

86

結合問合せによる欠損(1)

- 表deptからDatabase部門を削除した表dept2を考える。

dept2

deptno	dname	manager
D01	Account	E01
D02	Personnel	E04

- 表empと表dept2を次の結合問合せで結合すると、どのような結果が得られるか？

```
select a.ename, b.dname from emp a join dept2 b
   on a.deptno = b.deptno
```

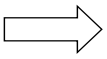
87

結合問合せによる欠損(2)

- 表empを、表dept2と結合すると、表deptと結合したときと比べて**タップルが欠損する**

```
select a.ename, b.dname
from emp a join dept b
on a.deptno = b.deptno
```

ename	dname
Smith	Account
Morgan	Account
Robert	Account
Washington	Personnel
Lincoln	Personnel
Taylor	Account
Suzuki	**Database**
Tanaka	**Database**
Matsuda	**Database**



```
select a.ename, b.dname
from emp a join dept2 b
on a.deptno = b.deptno
```

ename	dname
Smith	Account
Morgan	Account
Robert	Account
Washington	Personnel
Lincoln	Personnel
Taylor	Account

88

外結合(outer join)

- 結合問合せで、結合条件で指定された列間で、一方の表にある値が他方の表にはない場合、検索条件不成立となり、タップルの欠損が生じる
- このような時でも、**外結合(outer join)**を行うことにより、存在しないところは空値にして結合結果を出すことで、元の表からのタップルの欠損を防ぐことができる (SQL:1999で導入された構文）
- 外結合には、次の3種類がある
  - **左外結合(left outer join)**：結合する左側の表の欠損を防ぐ
  - **右外結合(right outer join)**：結合する右側の表の欠損を防ぐ
  - **完全外結合(full outer join):** 結合する両側の表の欠損を防ぐ

89

外結合の例

- 表empと表dept2を**左外結合**（または**完全外結合**）により結合するとタップルの欠損は起こらない。

```
select a.ename, b.dname
     from emp a left outer join dept2 b
     on a.deptno = b.deptno
```

または、

```
select a.ename, b.dname
     from emp a full outer join dept2 b
     on a.deptno = b.deptno
```

ename	dname
Smith	Account
Morgan	Account
Robert	Account
Washington	Personnel
Lincoln	Personnel
Taylor	Account
Suzuki	**NULL**
Tanaka	**NULL**
Matsuda	**NULL**

NULLは空値を表す

参考：NULLはシステム内部の表現であり、実行結果では**表示されない**（空白となる）