Databases - 4

Other relational operations and DDL

How to write RA expressions for dummies

Step 1: Identify the relations required and CP them together

Step 2: Add required selections to make the CP into an appropriate Join

Step 3: Add any other selections required for the query

Step 4: Add appropriate projections to get the required attributes for the query

RA example

Show the name, job, sal and location for all staff with a salary greater than £25,000

How to write RA expressions for dummies

Step 1: Identify the relations required and CP them together

Step 2: Add required selections to make the CP into an appropriate Join

Step 3: Add any other selections required for the query

Step 4: Add appropriate projections to get the required attributes for the query

RA example

Show the name, job and location for all staff with a salary greater than £25,000

Step 1: Identify the relations required and CP them together

Requires 2 tables EMP and DEPT

$emp \times dept$

How to write RA expressions for dummies

Step 1: Identify the relations required and CP them together

Step 2: Add required selections to make the CP into an appropriate Join

Step 3: Add any other selections required for the query

Step 4: Add appropriate projections to get the required attributes for the query

RA example

Show the name, job and location for all staff with a salary greater than £25,000

Step 2: Add required selections to make the CP into an appropriate Join

$$\sigma_{\text{predicate}}(\text{emp} \times \text{dept})$$

$$\sigma_{\text{emp.deptno=dept.deptno}}(\text{emp} \times \text{dept})$$

How to write RA expressions for dummies

Step 1: Identify the relations required and CP them together

Step 2: Add required selections to make the CP into an appropriate Join

Step 3: Add any other selections required for the query

Step 4: Add appropriate projections to get the required attributes for the query

RA example

Show the name, job and location for all staff with a salary greater than £25,000

Step 3: Add any other selections required for the query

$$\sigma_{\text{emp.deptno=dept.deptno}}(\text{emp} \times \text{dept})$$

RA example

Show the name, job and location for all staff with a salary greater than £25,000

Step 3: Add any other selections required for the query

$$\sigma_{emp.deptno=dept.deptno}(emp \times dept)$$

$$\sigma_{sal>25000}(\sigma_{emp.deptno=dept.deptno}(emp \times dept))$$

Note

Alternatives here: could simply use an AND

$$\sigma_{sal>25000}(\sigma_{emp.deptno=dept.deptno}(emp \times dept))$$

Would be the same as

$$\sigma_{(emp.deptno=dept.deptno)\ AND\ sal>25000}(emp \times dept)$$

How to write RA expressions for dummies

Step 1: Identify the relations required and CP them together

Step 2: Add required selections to make the CP into an appropriate Join

Step 3: Add any other selections required for the query

Step 4: Add appropriate projections to get the required attributes for the query

RA example

Show the **name**, **job** and **location** for all staff with a salary greater than £25,000

Step 4: Add appropriate projections to get the required attributes for the query

$$\sigma_{sal>25000}(\sigma_{emp.deptno=dept.deptno}(emp \times dept))$$

$$\pi_{ename, job, loc}(\sigma_{sal>25000}(\sigma_{emp.deptno=dept.deptno}(emp \times dept)))$$

Important

Watch out for projections BEFORE selections - check they still work

$$\sigma_{sal>25000}(\pi_{ename, job, loc}(\sigma_{emp.deptno=dept.deptno}(emp \times dept)))$$

This produces an empty set (or an error result). Why?

How to write SQL expressions for dummies

Step 1: Identify the tables required and CP them together

Step 2: Add required conditions to make the CP into an appropriate Join

Step 3: Add any other conditions required for the query

Step 4: Add appropriate projections to get the required columns for the query

SQL example

Show the name, job, sal and location for all staff with a salary greater than £25,000

How to write SQL expressions for dummies

Step 1: Identify the tables required and CP them together

Step 2: Add required conditions to make the CP into an appropriate Join

Step 3: Add any other conditions required for the query

Step 4: Add appropriate projections to get the required columns for the query

SQL example

Show the name, job and location for all staff with a salary greater than £25,000

Step 1: Identify the tables required and CP them together

Requires 2 tables EMP and DEPT

```
select * or expression          projections here
from emp, dept                  CPs here
[where expression]              selections here
```

How to write SQL expressions for dummies

Step 1: Identify the tables required and CP them together

Step 2: Add required conditions to make the CP into an appropriate Join

Step 3: Add any other conditions required for the query

Step 4: Add appropriate projections to get the required columns for the query

SQL example

Show the name, job and location for all staff with a salary greater than £25,000

Step 2: Add required conditions to make the CP into an appropriate Join

```
select * or expression          projections here
from emp, dept                  CPs here
where emp.deptno=dept.deptno    selections here
```

How to write SQL expressions for dummies

Step 1: Identify the tables required and CP them together

Step 2: Add required conditions to make the CP into an appropriate Join

Step 3: Add any other conditions required for the query

Step 4: Add appropriate projections to get the required columns for the query

SQL example

Show the name, job and location for all staff with a salary greater than £25,000

Step 3: Add any other conditions required for the query

```
select * or expression                  projections here
from emp, dept                          CPs here
where emp.deptno=dept.deptno            selections here
      and sal > 25000
```

How to write SQL expressions for dummies

Step 1: Identify the tables required and CP them together

Step 2: Add required conditions to make the CP into an appropriate Join

Step 3: Add any other conditions required for the query

Step 4: Add appropriate projections to get the required columns for the query

SQL example

Show the name, job and location for all staff with a salary greater than £25,000

Step 4: Add appropriate projections to get the required columns for the query

```
select ename, job, loc                  projections here
from emp, dept                          CPs here
where emp.deptno=dept.deptno            selections here
      and sal > 25000
```

Relational Algebra operations

Selection	σ
Projection	π
Cartesian Product	×
Union	∪
Set Difference	-
Join	⋈
Intersection	∩
Division	÷

Find all values

UNION

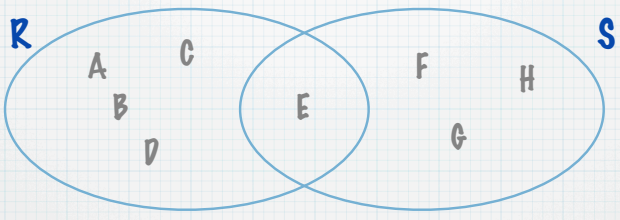
R ∪ S

A set, every member of which is an element of one or another of two or more given sets.

All the distinct values from the 'first' set along with all the distinct values in the 'second' set

UNION

R ∪ S



R ∪ S = { A, B, C, D, E, F, G, H }

RA example

Create a list of all the hisals and losals

list of all the hisals

$\pi_{hisals}(grade)$

hisal
21999
23999
29999
49999
99999

RA example

Create a list of all the hisals and losals

list of all the losals

$\pi_{losals}(grade)$

losal
17000
22000
24000
30000
50000

RA example

Create a list of all the hisals and losals

$\pi_{losals}(grade)$

$\cup$

$\pi_{hisals}(grade)$

21999
23999
29999
49999
99999
17000
22000
24000
30000
50000

SQL example

Create a list of all the hisals and losals

```
select hisal from grade
  union
select losal from grade
```

21999
23999
29999
49999
99999
17000
22000
24000
30000
50000

Relational Algebra operations

Selection	σ
Projection	π
Cartesian Product	×
Union	∪
Set Difference	-
Join	⋈
Intersection	∩
Division	÷

Find values in one set but not in another

SET DIFFERENCE

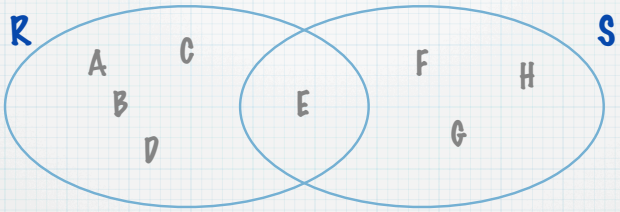
R - S

A set, every member of which is in one particular set but not another

All the distinct values from the 'first' set without all the distinct values in the 'second' set

SET DIFFERENCE

R - S



R - S = { A, B, C, D }

RA example

Create a list of actual salaries excluding any that are earning the highest value for the grade

list of all the salaries

$\pi_{sal}(emp)$

sal
26000
22500
39750
22500
38500
34500
40000
60000
25000
21000
19500
40000
23000
18000

RA example

Create a list of actual salaries excluding any that are earning the highest value for the grade

list of all the highest values

$\pi_{hisal}(grade)$

hisal
21999
23999
29999
49999
99999

RA example

Create a list of actual salaries excluding any that are earning the highest value for the grade

$$\pi_{sal}(emp) - \pi_{hisals}(grade)$$

Done using the SQL keyword **minus** - unfortunately not many PC databases support this syntax

RA example

So a cheat in SQL

```
select sal from emp
left join grade
on emp.sal=grade.hisal
where grade.hisal IS NULL
```

sal
26000
22500
39750
22500
38500
34500
40000
60000
25000
21000
19500
40000
23000
18000

(I will leave you to figure this one out)

Relational Algebra operations

Selection	σ
Projection	π
Cartesian Product	$\times$
Union	$\cup$
Set Difference	-
Join	$\bowtie$
Intersection	$\cap$
Division	$\div$

Done using CP and selection

Relational Algebra operations

Selection	σ
Projection	π
Cartesian Product	×
Union	∪
Set Difference	-
Join	⋈
Intersection	∩
Division	÷

Not really implemented in databases as can be done by selection

Relational Algebra operations

Selection	σ
Projection	π
Cartesian Product	×
Union	∪
Set Difference	-
Join	⋈
Intersection	∩
Division	÷

Not really implemented in databases as hard to implement

Data Query Language (DQL)

The SQL shown so far is for writing queries - the DQL part of the language

```
select * or expression
from relations
[where expression]
```

DDL and DML

SQL also has a syntax for creating tables, altering the structure of tables and deleting tables - called the Data Definition Language (DDL)

SQL also has a syntax for inserting rows, updating and deleting rows - called the Data Manipulation Language (DML)

Data Definition Language (DDL)

Creating a table

```
CREATE TABLE tablename
   (column_name   type [NULL/NOT NULL],
    column_name   type [NULL/NOT NULL],
      ..)
```

Data Definition Language (DDL)

type(s) - really depends on the actual database

CHAR (size)	Character data, maximum of 'size' characters upto 255
DATE	Dates (which include time)
TEXT	Character data up to 65535
INT	Whole numbers

... but lots of other ones as well..

Numbers ...

Numeric {1 byte}	TINYINT[(M)]	-128 TO 127 [0 to 255 if UNSIGNED]	AUTO_INCREMENT UNSIGNED, ZEROFILL, SERIAL DEFAULT VALUE	NULL [0 if NOT NULL]
Numeric {2 bytes}	SMALLINT[(M)]	-32,768 to 32,767 [0 to 65,535]	AUTO_INCREMENT, UNSIGNED, ZEROFILL, SERIAL DEFAULT VALUE	NULL [0 if NOT NULL]
Numeric {3 bytes}	MEDIUMINT[(M)]	-8,388,608 to 8,388,607 [0 to 16,777,215]	AUTO_INCREMENT, UNSIGNED, ZEROFILL, SERIAL DEFAULT VALUE	NULL [0 if NOT NULL]
Numeric {4 bytes}	INT[(M)]	-/+2.147E+9 [0 to 4.294E+9]	AUTO_INCREMENT, UNSIGNED, ZEROFILL, SERIAL DEFAULT VALUE	NULL [0 if NOT NULL]
Numeric {8 bytes}	BIGINT[(M)]	-/+9.223E+18 [0 to 18.45E+18]	AUTO_INCREMENT, UNSIGNED, ZEROFILL, SERIAL DEFAULT VALUE	NULL [0 if NOT NULL]
Numeric {4 or 8}	FLOAT(p)	p=0-24 --> "FLOAT" p=25-53 --> "DOUBLE"	UNSIGNED, ZEROFILL	NULL [0 if NOT NULL]
Numeric {4 bytes}	FLOAT[(M,D)]	Min=+/-1.175E-38 Max=+/-3.403E+38	UNSIGNED, ZEROFILL	NULL [0 if NOT NULL]
Numeric {8 bytes}	DOUBLE[(M,D)]	Min=+/-2.225E-308 Max=+/-1.798E+308	UNSIGNED, ZEROFILL	NULL [0 if NOT NULL]
Numeric {M+2}	DECIMAL[(M,[D])] Stored as string	Max Range = DOUBLE range Fixed point vs. DOUBLE float	UNSIGNED, ZEROFILL	NULL [0 if NOT NULL]

Binary patterns ...

Bit {8 bytes}	BIT[(M)]	Binary. Display by [add zero or converting with BIN()]. M=1-64	Prior to 5.03 TINYINT(1) Synonym	NULL [0 if NOT NULL]

Strings ...

String {M char's}	CHAR[(M)]	M=0-255 Characters, FIXED. Right padded with spaces.	BINARY, CHARACTER SET	NULL ["" if NOT NULL]
String {M char's[1]}	VARCHAR(M)	M=0-65,535 Characters M=0-255 <v5.0.3	BINARY, CHARACTER SET	NULL ["" if NOT NULL]
String {#char's[1]}	TINYTEXT[2]	0-255 Characters	BINARY, CHARACTER SET	NULL ["" if NOT NULL]
String {#char's[1]}	TEXT[2]	0-65,535 Char's	BINARY, CHARACTER SET	NULL ["" if NOT NULL]
String {#char's[1]}	MEDIUMTEXT[2]	0-16,777,215 Char's	BINARY, CHARACTER SET	NULL ["" if NOT NULL]
String {#char's[1]}	LONGTEXT[2]	0-4,294,967,295 Char's	BINARY, CHARACTER SET	NULL ["" if NOT NULL]

Large binary fields ...

String {#bytes[1]}	TINYBLOB	0-255 bytes	Global Only (case sensitive)	NULL ["" if NOT NULL]
String {#bytes[1]}	BLOB	0-65,535 bytes	Global Only (case sensitive)	NULL ["" if NOT NULL]
String {#bytes[1]}	MEDIUMBLOB	0-16,777,215 bytes	Global Only (case sensitive)	NULL ["" if NOT NULL]
String {#bytes[1]}	LONGBLOB	0-4,294,967,295 bytes	Global Only (case sensitive)	NULL ["" if NOT NULL]

Pictures, Sounds, Video - watch out using these

Sets ...

String {1-2 bytes}	ENUM[2] ("A1","A2",...)	Column is exactly 1 of 1-65,535 values	CHARACTER SET	NULL [1st value if NOT NULL]
String {1-8 bytes}	SET[2] ("A1","A2",...)	Column is 0 or more values in list of 1-64 members	CHARACTER SET	NULL ["" if NOT NULL]

I.e. collections where the values are known in advance and come from a limited range

manager, salesperson, administrator

Dates/Times ...

Date & Time {3 bytes}	DATE	"1000-01-01" - "9999-12-31"	Global Only (YYYY-MM-DD)	NULL ["0000-00-00" if NOT NULL]
Date & Time {8 bytes}	DATETIME	"1000-01-01 00:00:00" - "9999-12-31 23:59:59"	Global Only (YYYY-MM-DD hh:mm:ss)	NULL ["0000-00-00 00:00:00" if NOT NULL]
Date & Time {3 bytes}	TIME	"-838:59:59" - "838:59:59"	Global Only (hh:mm:ss)	NULL ["00:00:00" if NOT NULL]
Date & Time {4 bytes}	TIMESTAMP	19700101000000 - 2037+	Global Only (YYYYMMDDhhmmss)	Current Date & Time
Date & Time {1 bytes}	YEAR	1900 - 2155	Global Only (YYYY)	NULL ["0000" if NOT NULL]

Data Definition Language (DDL)

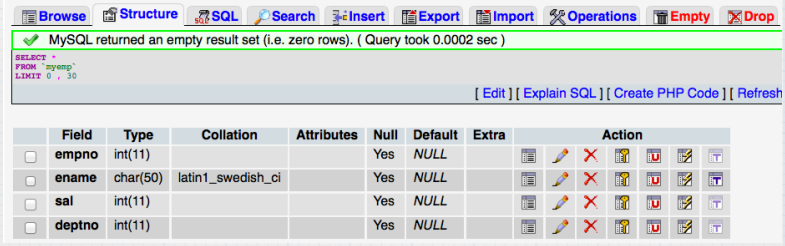
Creating a table

```
CREATE TABLE tablename
   (column_name   type [NULL/NOT NULL],
    column_name   type [NULL/NOT NULL],
      ..)
```

Data Definition Language (DDL)

Creating a table

```
CREATE TABLE myemp (
  empno int(1 1),
  ename char(50),
  sal int(1 1),
  deptno int(1 1)
)
```



Data Manipulation Language (DML)

Inserting a record

```
insert into table [(columnname, columnname, ...)]
   values (value, value,...)

insert into table
   values (value, value,...)
```

Data Manipulation Language (DML)

Inserting a record

```
insert into myemp (empno, ename, sal, deptno)
values (12, "smith", 25000, 3)
```

empno	ename	sal	deptno
12	smith	25000	3

Data Manipulation Language (DML)

Inserting a record (if inserting all values in a row)

```
insert into myemp
values (13, "jones", 24000, 2)
```

empno	ename	sal	deptno
12	smith	25000	3
13	jones	24000	2

Data Manipulation Language (DML)

Inserting records from other tables

```
insert into table [(columnname, columnname, ...)]
    select  expression
```

Data Manipulation Language (DML)

Copies records from another table

```
insert into myemp

select empno,
ename,sal,deptno
from emp
where sal<25000
```

empno	ename	sal	deptno
12	smith	25000	3
13	jones	24000	2
557	BELL	22500	3
690	AHMAD	22500	3
912	HAYES	21000	2
936	CASSY	19500	3
970	BLACK	23000	1
405	MARCH	18000	2

Data Manipulation Language (DML)

Updating record(s)

```
update tablename set name = value [(,name=value)]
[where expression]
```

Data Manipulation Language (DML)

Updating record(s)

```
update myemp set deptno = 4
where ename = "Bell"
```

empno	ename	sal	deptno
12	smith	25000	3
13	jones	24000	2
557	BELL	22500	4
690	AHMAD	22500	3
912	HAYES	21000	2
936	CASSY	19500	3
970	BLACK	23000	1
405	MARCH	18000	2

Data Manipulation Language (DML)

Updating record(s)

```
update myemp set sal = sal * 1.1
where ename = "Smith"
or ename = "Jones"
```

empno	ename	sal	deptno
12	smith	27500	3
13	jones	26400	2
557	BELL	22500	4
690	AHMAD	22500	3
912	HAYES	21000	2
936	CASSY	19500	3
970	BLACK	23000	1
405	MARCH	18000	2

Data Manipulation Language (DML)

Deleting record(s)

```
delete from tablename
[where expression]
```

Data Manipulation Language (DML)

Deleting Pollard

```
delete from myemp
where ename = "Jones"
```

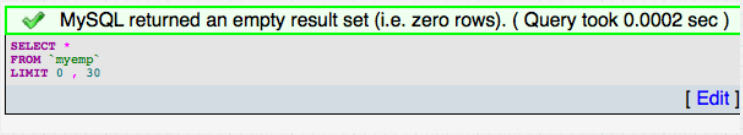
empno	ename	sal	deptno
12	smith	27500	3
13	jones	26400	2
557	BELL	22500	4
690	AHMAD	22500	3
912	HAYES	21000	2
936	CASSY	19500	3
970	BLACK	23000	1
405	MARCH	18000	2

empno	ename	sal	deptno
12	smith	27500	3
557	BELL	22500	4
690	AHMAD	22500	3
912	HAYES	21000	2
936	CASSY	19500	3
970	BLACK	23000	1
405	MARCH	18000	2

Data Manipulation Language (DML)

Watch out:
Deleting everything

```
delete from myemp
```



Data Definition Language (DDL)

Deleting a table - we say "dropping a table"

```
DROP TABLE tablename
```

```
drop table myemp
```