Databases

Andrzej Grzybowski

Institute of Physics, University of Silesia

Lecture 1

Relational Database Algebra

as a theoretical basis for

Structured Query Language
implemented in

Oracle Database Management System

Relational Database Algebra

In set theory a relation *R* :

- **over the set *A*** is any subset of the cartesian product $A \times A$: $R \subset A \times A$
- **over the sets *A* and *B*** is any subset of the cartesian product A×B: $R \subset A \times B$
- **over the sets $A_1, A_2, \ldots, A_n$** is any subset of the cartesian product $A_1 \times A_2 \times \ldots \times A_n$: $R \subset A_1 \times A_2 \times \ldots \times A_n$

Relational Database Algebra

- **A relation R in the relational database** (**RDB**) can be preliminarily treated from the viewpoint of set theory as any named subset of cartesian product of the sets $D_1, D_2, \ldots, D_n$, being domains (sets of values) of attributes $A_1, A_2, \ldots, A_n$, respectively.
- However, it should be noted that the use of cartesian product $D_1 \times D_2 \times \ldots \times D_n$ in the above definition is not precise. It is mainly because an order of the attributes $A_1, A_2, \ldots, A_n$ of RDB relation does not matter.

Relational Database Algebra

Tabular representation of RDB relation

R ← **relation name**

A	B	C
a1	b2	c3
a2	b1	c3
a3	b1	c2
a2	b2	c2

← relation header consists of attribute names

← tuple

← tuple

← tuple

← tuple

Attribute domains:

$D_A = D(A) = \{a1, a2, a3\}$

$D_B = D(B) = \{b1, b2\}$

$D_C = D(C) = \{c2, c3\}$

Relational Database Algebra

Required features for a relation R to be RDB relation:

1) atomicity: elements of tuples of relation R represent single values,
2) each attribute A has its distinguishable (unique) name,
3) all values of attribute A come from the same domain D,
4) an order of relation attributes do not matter,
5) each tuple is different (unique),
6) an order of relation tuples do not matter.

Compatibility of relations

- Schema of relation (in other words header of relation) R is a set of attribute names $A_1, A_2, \ldots, A_n$ of relation R:

  $N(R)=\{A_1, A_2, \ldots, A_n\}$
- Relations R and S are called compatible when they have the same headers: $N(R)=N(S)$

  tzn. $D(R.A_1)=D(S.A_1)$, $D(R.A_2)=D(S.A_2)$, …, $D(R.A_n)=D(S.A_n)$

RDB algebra operations

Based on set theory:

1) Union
2) Intersect
3) Difference
4) Cartesian product

Using information on structure of tuples:

5) Alias (renaming)
6) Projection
7) Selection
8) Joining
9) Division

Union

Let *R* and *S* are compatible relations.
Union of the relations *R* and *S* is
the relation ***R*** ∪ ***S***, having the same header,
that contains all tuples of *R*, *S*, or both the relations.

R:

X	Y	Z
x1	y1	z1
x1	y2	z3
x2	y1	z2

S:

X	Y	Z
x1	y1	z1
x1	y1	z2
x1	y2	z3
x3	y2	z3

R∪*S*:

X	Y	Z
x1	y1	z1
x1	y2	z3
x2	y1	z2
x1	y1	z2
x3	y2	z3

Standard SQL & Oracle SQL:

```
select X,Y,Z from R
union
select X,Y,Z from S;
```

Intersection

Let *R* and *S* are compatible relations.
Intersection of the relations *R* and *S* is the relation ***R ∩ S***, having the same header, that contains all tuples of both *R* and *S*.

R:

X	Y	Z
x1	y1	z1
x1	y2	z3
x2	y1	z2

S:

X	Y	Z
x1	y1	z1
x1	y1	z2
x1	y2	z3
x3	y2	z3

R∩S:

X	Y	Z
x1	y1	z1
x1	y2	z3

Standard SQL & Oracle SQL:

```
select X,Y,Z from R
intersect
select X,Y,Z from S;
```

Difference

Let *R* and *S* are compatible relations.
Difference of the relations *R* and *S* is
the relation ***R – S***, having the same header,
that contains all tuples of *R*, which are not in *S*.

R:

X	Y	Z
x1	y1	z1
x1	y2	z3
x2	y1	z2

S:

X	Y	Z
x1	y1	z1
x1	y1	z2
x1	y2	z3
x3	y2	z3

R–S:

X	Y	Z
x2	y1	z2

Standard SQL & Oracle SQL:

```
select X,Y,Z from R
minus
select X,Y,Z from S;
```

Cartesian product

Let relations *R* and *S* have headers $N(R)=\{X1,X2,...,Xn\}$ and $N(S)=\{Y1,Y2,...,Ym\}$, respectively .

Cartesian product of the relations *R* and *S* is the relation $R \times S$, having the header $N(R \times S)= \{R.X1,R.X2,...,R.Xn, S.Y1,S.Y2,...,S.Ym\}$, that contains tuples being all possible combinations of tuples from the relations *R* and *S*.

If attribute names of the relations R and S are different, then the cartesian product header can be written as:

$N(R \times S)= \{X1,X2,...,Xn,Y1,Y2,...,Ym\}$

Cartesian product

R:

X	Y	Z
x1	y1	z1
x1	y2	z3
x2	y1	z2

S:

X	Y	Z
x1	y1	z1
x1	y1	z2
x1	y2	z3
x3	y2	z3

R×S:

R.X	R.Y	R.Z	S.X	S.Y	S.Z
x1	y1	z1	x1	y1	z1
x1	y1	z1	x1	y1	z2
x1	y1	z1	x1	y2	z3
x1	y1	z1	x3	y2	z3
x1	y2	z3	x1	y1	z1
x1	y2	z3	x1	y1	z2
x1	y2	z3	x1	y2	z3
x1	y2	z3	x3	y2	z3
x2	y1	z2	x1	y1	z1
x2	y1	z2	x1	y1	z2
x2	y1	z2	x1	y2	z3
x2	y1	z2	x3	y2	z3

Standard SQL & Oracle SQL:

```
select R.X,R.Y,R.Z,S.X,S.Y,S.Z from R, S;
```

Relation alias

Alias of the relation *R* is called the relation renaming, for instance R=A

A:

X	Y	Z
x1	y1	z1
x1	y2	z3
x2	y1	z2

Oracle SQL:

select A.X,A.Y,A.Z from R A;

An example of neccessary application of alias to calculate the Cartesian square $R \times R$:

$R=A$, $R=B$, then:

$R \times R = A \times B$:

A.X	A.Y	A.Z	B.X	B.Y	B.Z
x1	y1	z1	x1	y1	z1
x1	y1	z1	x1	y2	z3
x1	y1	z1	x2	y1	z2
x1	y2	z3	x1	y1	z1
x1	y2	z3	x1	y2	z3
x1	y2	z3	x2	y1	z2
x2	y1	z2	x1	y1	z1
x2	y1	z2	x1	y2	z3
x2	y1	z2	x2	y1	z2

Attribute alias

Alias of the attribute X of the relation R is called the attribute renaming, for instance X=Xnowy

R:

Xnowy	Y	Z
x1	y1	z1
x1	y2	z3
x2	y1	z2

Oracle SQL:

select X as Xnowy,Y,Z from R;

lub

select X Xnowy,Y,Z from R;

Projection

Let the relation R has its header N(R)={$X1$,$X2$,...,Xn}. Projection of the relation R on its attributes $Xi1$,$Xi2$,...,Xik, where {$Xi1$,$Xi2$,...,Xik} $\subset$ {$X1$,$X2$,...,Xn}, is the relation T=R[$Xi1$,$Xi2$,...,Xik], having the header N(T)= {$Xi1$,$Xi2$,...,Xik}, that contains tuples t, meeting the following condition:

$\bigwedge$tuples $r \in R$ $\bigvee$ a single tuple $t \in T$:

$t(Xij)=r(Xij)$ $\bigwedge$ j: $1 \le j \le k$.

Another notation frequently used for projection of the relation R on its attributes $Xi1$,$Xi2$,...,Xik: $\pi_{Xi1,Xi2,...,Xik}(R)$

Projection

S:

X	Y	Z
x1	y1	z1
x1	y1	z2
x1	y2	z3
x3	y2	z3

S[X,Y]:

X	Y
x1	y1
x1	y2
x3	y2

Standard SQL & Oracle SQL:

select X,Y from S;

Selection

Let the relation *R* has its header N(*R*)={*X1*,*X2*,...,*Xn*}. Selection of the relation *R* according to the criterion *C* is the relation *T* = *R* where *C*, having the same header as the relation *R*, that contains tuples *s*, meeting the selection condition *C* which can be in forms as follows:

1) C can be any comparison:
   Xi $\propto$ Xj or Xi $\propto$ x, where x $\in$ D(Xi) – attribute domains *Xi*, and the comparison operator $\propto$ can be as follows: <, >, =, <=, >=, <>; another comparison operator of difference can also be used in Oracle: !=
2) If C' i C" are selection conditions, then a new condition C in the following forms: C' and C", C' or C", not C.

Another notation frequently used for selection of the relation *R* according to the criterion *C*: $\sigma_C(R)$

Selection

R:

X	Y	Z
x1	y1	z1
x1	y2	z3
x2	y1	z2

R where (*X*=x1 and *Y*=y2)

X	Y	Z
x1	y2	z3

Standard SQL i Oracle SQL:

```
select X,Y,Z from R
where X=x1 and Y=y2;
```

Joins

- Natural joining (refers usually to identically named attributes of joined relations)
- Inner joining (also called: simple joining), its result are tuples, exactly meeting the join condition
- Outer joining (superset of inner joining):
  - left outer joining,
  - right outer joining,
  - full outer joining
- Equi-joining (to compare shared attributes we use the operator =)
- θ-join,(to compare shared attributes we use the operator θ)
  - if the operator θ is: <, >, <=, >=, then the joining is called semi-joining
  - if the operatore θ is the comparison operator of difference <> (!=), then the joining is called anti-joining
- Self-joining (joining a relation to itself)

Natural joining

Let the relation R and S have headers
$N(R)=\{X1,X2,...,Xn,Z1,Z2,...,Zk\}$ and
$N(S)=\{Z1,Z2,...,Zk,Y1,Y2,...,Ym\}$, respectively,
where $n,k,m \geq 0$ oraz $\{Z1,Z2,...,Zk\}$ is
a complete subset of attributes shared by both the relations.
Natural joining R and S is the relation $R \triangleright\triangleleft S$, having the header
$N(R \triangleright\triangleleft S) = \{X1,X2,...,Xn,Z1,Z2,...,Zk,Y1,Y2,...,Ym\}$, that contains only the tuples t which meet the condition of equi-joining:

$\bigvee$ tuples $r \in R$ as well as $s \in S$: $r(Zi)=s(Zi)$ $\bigwedge$ i: $1 \leq i \leq k$

then $\bigvee$ $t \in R \triangleright\triangleleft S$: $t(X1,X2,...,Xn,Z1,Z2,...,Zk,Y1,Y2,...,Ym)$

NATURAL JOINING belongs
to INNER JOINS as well as EQUI JOINS.
Sometimes it is equated simply with JOINING.

However, the most frequently, as it is for example in Oracle SQL, JOINING is equated with INNER JOINING!

Natural joining

R:

X	*Z1*	*Z2*
x1	z1	z1’
x1	z2	z1’
x2	z1	z2’

S:

Z1	*Z2*	*Y*
z1	z1’	y1
z1	z1’	y2
z1	z2’	y3
z2	z2’	y4

$R \mid\!\bowtie\!\mid S$:

X	*Z1*	*Z2*	*Y*
x1	z1	z1’	y1
x1	z1	z1’	y2
x2	z1	z2’	y3

Natural joining

Only opportunity with older versions of Oracle SQL:

```
select X, R.Z1, R.Z2, Y from R, S
where R.Z1=S.Z1 and R.Z2=S.Z2;
```

New opportunities with Oracle SQL inspired by Standard SQL:

```
select X, Z1, Z2, Y from R natural join S;
select X, R.Z1, R.Z2, Y from R inner join S
on R.Z1=S.Z1 and R.Z2=S.Z2;
select X, R.Z1, R.Z2, Y from R join S
on R.Z1=S.Z1 and R.Z2=S.Z2;
```

Left outer joining

R:

X	*Z1*	*Z2*
x1	z1	z1’
x1	z2	z1’
x2	z1	z2’

S:

Z1	*Z2*	*Y*
z1	z1’	y1
z1	z1’	y2
z1	z2’	y3
z2	z2’	y4

$R \rhd\!\lhd|_{LO} S$:

X	*Z1*	*Z2*	*Y*
x1	z1	z1’	y1
x1	z1	z1’	y2
x2	z1	z2’	y3
x1	z2	z1’	NULL

Left outer joining

Only opportunity with older versions of Oracle SQL:

```
select X, R.Z1, R.Z2, Y from R, S
where R.Z1=S.Z1(+) and R.Z2=S.Z2(+);
```

New opportunities with Oracle SQL inspired by Standard SQL:

```
select X, R.Z1, R.Z2, Y from R left join S
on R.Z1=S.Z1 and R.Z2=S.Z2;
select X, R.Z1, R.Z2, Y from R left outer join S
on R.Z1=S.Z1 and R.Z2=S.Z2;
```

And even:

```
select X, Z1, Z2, Y from R natural left join S;
select X, Z1, Z2, Y from R natural left outer join S;
```

Right outer join

R:

X	*Z1*	*Z2*
x1	z1	z1'
x1	z2	z1'
x2	z1	z2'

S:

Z1	*Z2*	*Y*
z1	z1'	y1
z1	z1'	y2
z1	z2'	y3
z2	z2'	y4

$R \rtimes|_{RO} S$:

X	*Z1*	*Z2*	*Y*
x1	z1	z1'	y1
x1	z1	z1'	y2
x2	z1	z2'	y3
NULL	z2	z2'	y4

Right outer join

Only opportunity with older versions of Oracle SQL:

```
select X, R.Z1, R.Z2, Y from R, S
where R.Z1(+)=S.Z1 and R.Z2(+)=S.Z2;
```

New opportunities with Oracle SQL inspired by Standard SQL:

```
select X, R.Z1, R.Z2, Y from R right join S
on R.Z1=S.Z1 and R.Z2=S.Z2;
select X, R.Z1, R.Z2, Y from R right outer join S
on R.Z1=S.Z1 and R.Z2=S.Z2;
```

And even:

```
select X, Z1, Z2, Y from R natural right join S;
select X, Z1, Z2, Y from R natural right outer join S;
```

Full outer joining

R:

X	*Z1*	*Z2*
x1	z1	z1’
x1	z2	z1’
x2	z1	z2’

S:

Z1	*Z2*	*Y*
z1	z1’	y1
z1	z1’	y2
z1	z2’	y3
z2	z2’	y4

$R \rtimes\!\!\!\!\bowtie_{\mathrm{O}} S$:

X	*Z1*	*Z2*	*Y*
x1	z1	z1’	y1
x1	z1	z1’	y2
x2	z1	z2’	y3
x1	z2	z1’	NULL
NULL	z2	z2’	y4

Full outer joining

Only opportunity with older versions of Oracle SQL:

```
select X, R.Z1, R.Z2, Y from R, S
where R.Z1=S.Z1(+) and R.Z2=S.Z2(+)
union
select X, R.Z1, R.Z2, Y from R, S
where R.Z1(+)=S.Z1 and R.Z2(+)=S.Z2;
```

New opportunities with Oracle SQL inspired by Standard SQL:

```
select X, R.Z1, R.Z2, Y from R full join S
on R.Z1=S.Z1 and R.Z2=S.Z2;
select X, R.Z1, R.Z2, Y from R full outer join S
on R.Z1=S.Z1 and R.Z2=S.Z2;
```

And even:

```
select X, Z1, Z2, Y from R natural full join S;
select X, Z1, Z2, Y from R natural full outer join S;
```

Comment on implementation of natural joining in Oracle SQL

Natual joining relations in Oracle SQL is executed on shared attributes having the same names. If such attributes do not exist, then cartesian product of the relations is executed.

It is permitted to use natural joining also to execute outer joining relations. However, in order to obtain such a result, joined relations have to possess shared attributes having the same names, otherwise cartesian product of the relations will be executed.

Division

Let the relations R and S have headers $N(R)=\{X1,X2,...,Xn,Z1,Z2,...,Zk\}$ and $N(S)=\{Z1,Z2,...,Zk\}$, respectively. It means that $N(S) \subset N(R)$.

Relation $T = R \div S$, having its header $N(T)= \{X1,X2,...,Xn\}$, is a result of dzivision of R by S, if T contains exactly the tuples t which meet the following condition:

$$\bigvee R_S \subset R:\ |R_s| = |S| \wedge |R_s[X1,X2,...,Xn]|=1 \wedge$$

$$\wedge \left(\bigwedge \text{tuples } s \in S \bigvee \text{a tuple } r \in R_S: r(Zi)=s(Zi) \bigwedge i: 1 \le i \le k \right)$$

then $\bigvee$ a single tuple $t \in R \div S: t(X1,X2,...,Xn)$.

There are no division operator in Oracle SQL.

Division

R:

X	*Y*	*Z*
x1	y1	z1
x2	y1	z1
x1	y2	z1
x1	y2	z2
x2	y1	z2

S1:

Z
z1

R÷*S1*:

X	*Y*
x1	y1
x2	y1
x1	y2

S2:

Y	*Z*
y1	z1

R÷*S2*:

X
x1
x2

S3:

Y	*Z*
y1	z1
y2	z2

R÷*S3*:

X
x1

Useful properties

Associative law:

for cartesian product : $(R \times S) \times T = R \times (S \times T)$

for joining : $(R \rhd\lhd S) \rhd\lhd T = R \rhd\lhd (S \rhd\lhd T)$

Commutative law:

for cartesian product: $R \times S = S \times R$

for joining: $R \rhd\lhd S = S \rhd\lhd R$

PRIORITY OF THE OPERATOR ACTING

THE HIGHEST

THE LOWEST

PROJECTION
SELECTION
CARTESIAN PRODUCT
JOINING, DIVISION
DIFFERENCE
UNION, INTERSECT

For example: *R*[*X*,Y] where *Z=5*
is incorrect (a result of projection *R*[*X*,Y] has no attribute *Z*).
Correct notation is: (*R* gdzie *Z=5)*[*X*,Y]

Of course, it is sometimes necessary to use brackets.

Complete set of operations

A COMPLETE SET OF RELATIONAL DATABASE ALGEBRA OPERATIONS consists of:

- UNION
- DIFFERENCE
- CARTESIAN PRODUCT
- PROJECTION
- SELECTION
- ALIAS (RENAMING)

Using these 6 operations it is possible to express the other 3:

INTERSECTION, JOINING, DIVISION.