[Gautam, 4(9):September 2017]
DOI- 10.5281/zenodo.857430

ISSN 2348 – 8034
Impact Factor- 4.022

GLOBAL JOURNAL OF ENGINEERING SCIENCE AND RESEARCHES

KEY CONCEPT AND ANALYSIS OF RELATIONAL ALGEBRA

Bhawana Gautam*[1], Dr. Preeti Bala[2] & Harsh Srivastava[3]
*[1&3]M.Tech. Student, Centre for Computer Sci. & Tech, Central University of Punjab, Bathinda
[2]Visiting faculty in IMT-CDL Ghaziabad & IGNOU

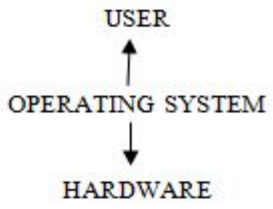
ABSTRACT

This paper represents the concept of keys and relational algebra in database management system. Basically key is an attribute that is able to identify the records in given relation uniquely. Relational algebra is procedural query language. It takes instance of relations as an input and gives the desired output in form of relation. For evaluating desired output there may be several queries in relational algebra but based on record comparisons and block retrievals efficiency of query is calculated and after identifying efficient query, that query is applied to find out the desired set of records in relational algebra.

Keywords: *Primary key, candidate keys, super keys, queries, integrity and relations*

I. INTRODUCTION

It may be defined as collection of related data, e.g. collection of student's details. Database management system is basically software, which is used to manage and access data in efficient way



Flat file system (OS files) was used to manage data beforedatabase management system came in existence. Flat file system was able to maintain data of less volume but if data size increases, this approach becomes irrelevant.

KB-----MB-----GB-----Above (TB)

Database is too large so that Flat File System fails to manage it.

Limitations of Flat File Systems are difficult to access required data. To avoid this, DBMS provides data independency (SQL Interface).It also takes more input/output cost, because required data may be stored in secondary memory and when data is needed it has to be brought into main memory from secondary memory. This procedure increases the i/o cost definitely. To overcome from this problem, DBMS uses indexing. Less degree of concurrency is also the limitations of flat file systems. DBMS provides high degree of concurrency (no. Of users accessing the same data simultaneously).This feature of database system avoids inconsistency.

II. INTEGRITY CONSTRAINTS

Integrity means something like 'be right', correctness and consistent. The data in a database must be right and in good condition. Because DBMS provides high degree of concurrency so that there is chances of integrity violation that is not desirable in database. In relational database data must be stored in tabular form.Integrity constraints provide a way of ensuring that changes made to the database by authorized users do not result in a loss of data consistency.An integrity constraint can be any arbitrary predicate applied to the database.

Sid	Sname	Branch
S1	A	CS
S2	B	EC
S3	C	EE
S4	D	CE
S5	E	CE

Arity:- Number of fields in table.

Cardinality:- Number of tuples in table.
So, arity of given table=3,
Cardinality of given table=5.

Relational Schema:- It means definition of table or structure of table i.e. how to define the table is known as relational schema.
Student(Sid, Sname , Branch).

Relational Instance:-Snapshots of table means record set i.e. no two records of database table should be same.

Candidate Key:-Purpose of this key is to provide uniqueness property in records of table.

Definition:- A candidate key is a column, or set of columns, in a table that can uniquely identify any database record without referring to any other data. Each table may have one or more candidate keys, but one candidate key is unique, and it is called the primary key i.e. minimal set of attributes used to differentiate records uniquely.
Student(Sid, Sname , Branch)

In the above example Sid is candidate key and also simple candidate key.

{Sid, Sname} is not candidate key, because it is able to differentiate records but not minimal.
Enroll(Sid, Cid, Fee)

Sid	Cid	Fee
S1	C1	10K
S2	C2	20K
S3	C3	20K

Simple Candidate Key:- Candidate key with single attribute.
{Sid} is Simple candidate key.

Complex(Compound) Candidate key:- Candidate key with more than one attribute.
{Sid, Cid} is Complex candidate key.

Candidate key integrity constraints are Uniqueness and Minimal.
{Cid, Fee} are alternative keys. Hence, a relation can have more than one candidate key. Out of all candidate keys, we can take one of candidate key as primary key.

Primary Key:-A primary key is a special relational database table column (or combination of columns) designated to uniquely identify all table records.It is one of candidate key whose field shouldn't be NULL value.
A primary key's main features are that it must contain a unique value for each row of data .It cannot contain null values.

[NULL: (unknown, unexisting)]

Alternative Key:- All candidate keys except primary key. In this Null values allowed. More than one alternative key is allowed to define in table.

Super Key:-A superkey is a combination of columns that uniquely identifies any row within a relational database management system (RDBMS) table. A candidate key is a closely related concept where the superkey is reduced to the minimum number of columns required to uniquely identify each row. Set of attributes used to differentiate record of table uniquely.

Hence, if we exclude "minimal"from candidate key definition then it turns to super key. Every candidate key is super key but not vice versa.

Foreign Key:- Foreign keys are the columns of a table that points to the primary key of another table. They act as a cross-reference between tables.

In other words, it is a set of attributes references to primary key or alternate key of same relation or other relation.

III. Relational Algebra

Relational algebra is a procedural query language. It takes instances of relations as input and gives instances of relations as output. In relational algebra, operators are used to perform queries. Operators may be unary or binary. There are some basic operations in relational algebra that are listed below:

Π: Projection – This operation is used to project those columns that satisfy the predicate.

σ: Selection – This selects tuples that satisfy given predicate from relation.

×: Cartesian product – It gives the all possible combinations of given two relations.

U: Union – It performs binary relations between two given relations.

-: Set – Result of this operation is tuples that are present in 1st relation but not in 2nd relation.

ῥ: Rename – Rename operation is performed in two ways. In 1st table will be renamed and in 2nd attributes will be renamed.

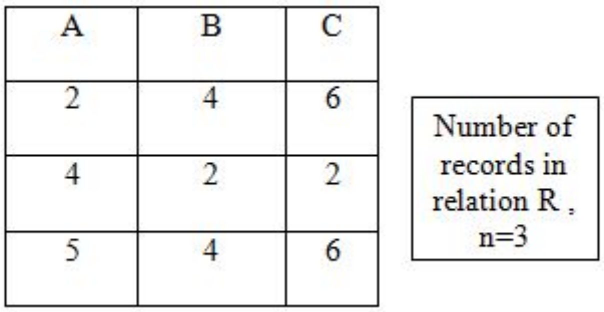
Basic operations of relational algebra means other operations can be derived from these operations like division (/), and intersection (∩).These operators are used to perform more complex queries

ISSN 2348 – 8034
Impact Factor- 4.022

IV. ANALYSIS OF RELATIONAL ALGEBRA OPERATIONS

a. Cartesian Product

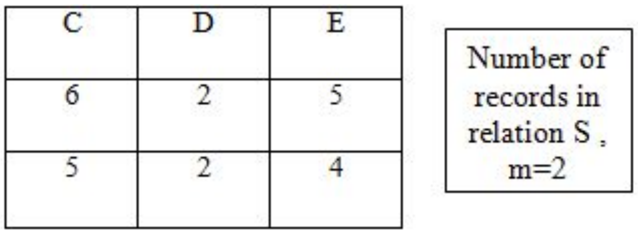
Relation R



A	B	C
2	4	6
4	2	2
5	4	6

Number of records in relation R , n=3

Relation S



C	D	E
6	2	5
5	2	4

Number of records in relation S , m=2

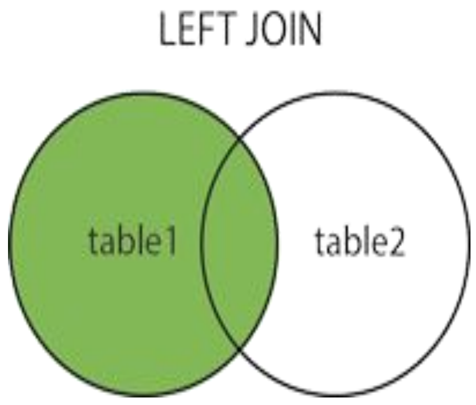
Now on performing Cartesian product operation on given relation R and S, the output will be R×S

A	B	C	C	D	E
2	4	6	6	2	5
2	4	6	5	2	4
4	2	2	6	2	5
4	2	2	5	2	4
5	4	6	6	2	5
5	4	6	5	2	5

Hence it can be seen that the cardinality of Cartesian product R×S is (n*m). So that for above example R×S is having 6 cardinality means number of records in output is 6.

b. Outer Join

- Left Outer Join: This operation gives the all records from left table and matches it with the right table. The result is NULL from right side if there is no match. In SQL it is called LEFT JOIN.



LEFT OUTER JOIN with given table R and S, the output will be:

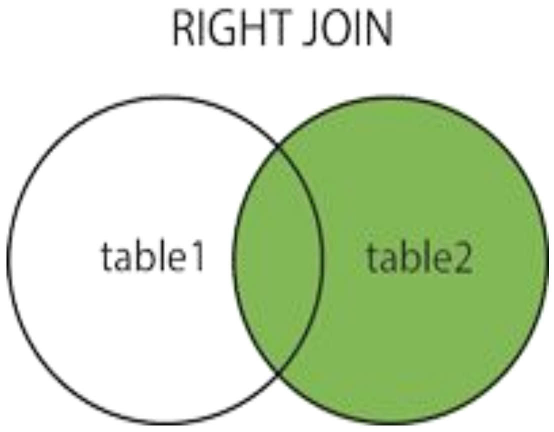
A	B	C	D	E
2	4	6	2	5
5	4	6	2	5
4	2	2	NULL	NULL

Cardinality ranges from (n) to (n*m), where n is cardinality of table R (Left table) and (m) is cardinality of S (Right table).

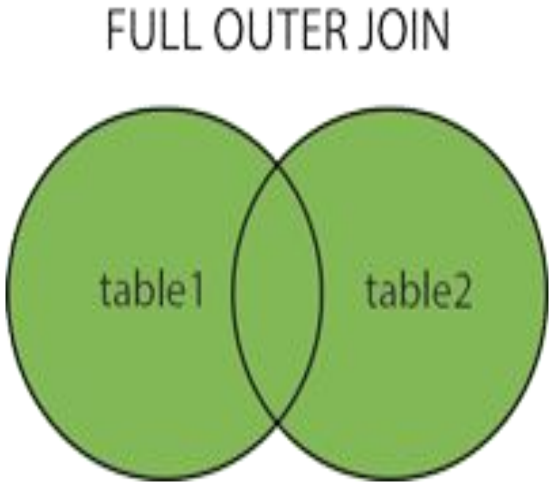
- Right Outer Join: This operation gives all records from right table and matches it with left table and results NULL from left side if there is no match.

RIGHT OUTER JOIN with given table R and S, the output will be:

A	B	C	D	E
2	4	6	2	5
5	4	6	2	5
NULL	NULL	5	2	4



- Full Outer Join: This operation gives the all records when there is match in either left table or in right table. FULL OUTER JOIN has potential to give very result sets.



Cardinality of FULL OUTER JOIN ranges from max (n,m) to (m*n) where (n) and (m) are the cardinalities of left table (R) and right table (S).

- Conditional Join (∞c): Conditional join operation is equivalent to conditional Cartesian product operation.

$$\boxed{R \infty_c S \equiv \sigma_c (R \times S)}$$

If two attributes are common and want to test equality only for one attribute then conditional join will be applied.

R (ABCD) and S (CDE)

If want to test only D equality then conditional join will be performed:

$$\boxed{R \infty_{(R.D = S.D)} S}$$

Here, relational schema is given to explain the operations of relational algebra:

Student (<u>Sid</u>, Sname, dob)
Course (<u>Cid</u>, Cname, Instructor)
Grades (<u>Sid Cid</u>, grade)

- To project Cid's that are taught by Korth, following operations will be performed:
  $$\pi Cid(\sigma Instructor = Korth(Course))$$

In above query, first of all 'Course' table will be selected by using select operator (σ) with condition that Instructor must be Korth after this from obtained rows only 'Cid' will be projected by using project operator (π).

- Retrieving student names who enrolled in at least one course.

$\pi Sname(\sigma Student.Sid=Grades.Sid(Student\times grades))$

In this query, first of all Cartesian product on table 'Student' and 'Grades' are performed after this select operator (σ) is used with condition that Sid of table 'Student' must be equal to 'Sid' of table 'Grades'. This results all rows that are having only those students who enrolled in at least one course. Finally, by using projection operator (π) 'Sname' column is projected.

- Analyzing efficiency of relational algebra queries in terms of number of comparisons when queries are equivalent.

To analyze this, one example is explained below:

- Retrieve student names from above schema who got 'A' grade in at least one course. Suppose 'Student' table is having 4 number of records and 'Grades' table is also having 4 number of records

Sid	Sname	Dob
S1	Ad	1/02/1990
S2	Ad	1/03/1990
S3	Bh	12/06/1993
S4	Ha	26/04/1991

[Student table]

Sid	Cid	Grade
S1	C1	A
S1	C2	B
S2	C2	A
S3	C1	B

[Grades table]

ISSN 2348 – 8034
Impact Factor- 4.022

$$\pi_{Sname}(\sigma_{Student.Sid=Grades.Sid \wedge grade = A}(Student \times Grades)) \text{ ----------- (i)}$$

$$\pi_{Sname}(\sigma_{Student.Sid=Grades.Sid}(Student \times (\sigma_{grade = A}(Grades)) \text{ -------------------- (ii)}$$

Query (i) and (ii) both are equivalent hence produces same result (Student name who got 'A' grade in at least one course).

But in terms of comparisons (i) is less efficient than (ii).

In query (i), firstly Cartesian product is performed in between table 'Student' and 'Grades' and 16 number of records are created (4*4). After this two conditions are checked for each record, 1st is 'Student.Sid' = 'Grades.Sid' and 2nd is 'Grade' = 'A'. Because in Cartesian product table, there is 16 records so that 16 comparisons to match first condition and 16 comparisons to match second condition.

So total number of comparisons in first approach for given tables are 32 (16+16).

In query (ii), firstly selection operator is applied on 'Grades' table with condition (grade = A), this gives filtered data means new Grades table with only 2 rows instead of 4. After this, Cartesian product is applied on table 'Student' and new filtered table 'Grades'. This produces 8 numbers of records (4*2). Finally only one condition is matched with each record of Cartesian table to filter out required data. In this approach total number of comparisons is 12 (8+4).

This analysis concludes that 2nd approach is more efficient because it requires only 12 comparisons whereas 1st requires 32 comparisons for same data set

- Student (<u>Sid</u>, Sname, Marks, Gender)

- Retrieving female students who scored more marks at least from one male student.

To perform this query, we need two instances of given 'Student' table. So that rename operator is also used

$$\dot{\rho}\ (T1, \sigma_{Gender=F}(Student)) \text{ ----- (i)}$$

$$\dot{\rho}\ (T1, \sigma_{Gender=M}(Student)) \text{ ----- (ii)}$$

Now Cartesian product is applied on table ‘T1’ and table ‘T2’ to get final output:

$$\boxed{\pi_{Sid}(\sigma_{T1.Marks > T2.Marks}(T1 \times T2))}$$

Sid	Sname	Marks	Gender
S1	A	10	F
S2	B	20	M
S3	C	30	F
S4	D	40	M
S5	E	50	F

[Student Table]

Set Operations: In relational algebra there are mainly three set operations: union (U), set difference (-) and intersection (∩).

To perform set operation on any given relations, say R and S must be union compatible, it means:

- Number of columns of ‘R’ and ‘S’ should be same.

Domain Ai should be similar to domain Bi.

[T1 (Sid, Sname) U T2 (Sid, Marks)]

$\swarrow$

[Union operation cannot be performed, not compatible]

R U S [OR]: {X/X € R V X € S}
R ∩ S [AND]: {X/X € R ^X € S}
R – S : It is similar to not operation means records present in ‘R’ should not present in ‘S’.

Division operation: This operation does not come under fundamental operations of relational algebra, it is derived operation and it is used when we want ‘every’ case record.

For example, retrieve Sid’s who enrolled in every course. To perform this first extract those students who are not enrolled in every course

ISSN 2348 – 8034
Impact Factor- 4.022

$$\boxed{\prod_{Sid} \left(\prod_{sid}(E) \times \prod_{Cid}(C) - E\right) \text{ ------------ (i)}}$$

This data is termed as disqualified data.After this perform set operation of this data with (∏sid(E)

$$\boxed{\left(\prod_{sid}(E) - \prod_{Sid} \left(\prod_{sid}(E) \times \prod_{Cid}(C) - E\right)\right.}$$

Relation 'R' with 'n' tuples.
Relation 'S' with 'm' tuples.
Cardinality of 'R/S' = {0 to n/m}.

V. Conclusion

From above explanation it can be concluded that key is the attribute which is used to identify records uniquely in any relation. Relational algebra explains how desired data is extracted from given relations, that's why relational algebra is also termed as procedural query language

REFERENCES

1. *'Database System Concepts',Sixth EditionAvi SilberschatzHenry F. KorthS. Sudarshan.*
2. *Rakesh Agrawal, Tomasz Imielinski, and Arun Swami. Database mining: A performance perspective. IEEE Transactions on Know ledge and Data Engineering, 5(6):914{925, December 1993.*
3. *Arbor Software Corporation, Sunnyvale, CA. Multidimensional Analysis: Converting Corporate Data into Strategic Informatio*
4. *Don Chamberlin. Using the New DB2: IBM's Object-Relational Database System. Morgan Kaufmann, 1996.*