The Relational Model

- Structure of Relational Databases
 - Mathematical Relations
 - Relational Schema & Relational Instances
 - Keys
- Relational Algebra
 - Relational Algebra Operations
 - Example queries

DBMS 2022 2.1

Mathematical Relations

- The relational model is built upon the concept of Mathematical Relations
- A relation is a (not necessarily proper) subset of a Cartesian product on n sets(Domains).

D1 x D 2x x Dn = { (d 1, d2,....dn) / d1
$$\epsilon$$
 D1, d2 ϵ D2.... dn ϵ Dn }

D1,D2...Dn are domains of the elements in the relation.

E.g.
$$\{(1, a), (1, e), (2, e)\} \le \{1, 2\} \times \{a, c, e\}$$

It is a relation on the sets $\{1, 2, g\}$ and $\{a, c, e\}$

2.3

The Relational Model

- Proposed by E.F.Codd in the early seventies.
- Most of the modern DBMS are relational.
- Simple and elegant model with mathematical basis.
- Led to the development of a theory of data dependencies and database design.
- Relational algebra operations –crucial role in query optimization & execution.
- Laid the foundation for the development of
 - Tuple relational calculus and then
 - Database standard SQL

DBMS 2022

Relation Schema

- The mathematical formality can be used in the database context to define relation schemas.
- A relation schema consists of relation name, and a set of attributes or field names or column names. Each attribute has an associated domain. Domain is the set of allowed values of the attribute.
- . $A_1, A_2, ..., A_n$ are attributes $R(A_1, A_2, ..., A_n)$ is a relation schema



When the domains are known, shorthand notation can be used:

Customer (customer-Id, customer-name, customer-street, customer-city)

2.4

Relation Instance

- An element of a relation is called a tuple.
- A relation instance is the collection of tuples of a relation at a particular moment in time. Relation instance of a relation are specified by a table. So tuples are rows in the table.
- Degree of a relation : The number of attributes (e.g. ---4)
- Cardinality of a relation : The number of tuples (e.g.---4)

		customer			attributes or columns)
cust	omer-ID	customer-name	customer-street	Category	ĺ
57 57	00893 00379 00321 00990	Jones Smith Curry Lindsay	Main NULL North NULL	A A C B	tuples (or rows)

Attribute Types

- Attribute values are (normally) required to be atomic, that is, indivisible
 - Multi valued attribute values are not atomic (e.g. phonenumber)
 - composite attribute values are not atomic (e.g. address)
- The special value *null* is a member of every domain. It represent the unknown values or unspecified values.

DBMS 2022 JCJ DBMS 2022

Relations are Unordered

2.5

- Order of tuples is irrelevant (tuples may be stored in an arbitrary order)
- The following two tables represent the *same* relation:

account-number	branch-name	balance
A-101	Downtown	500
A-102	Perryridge	400
A-201	Brighton	900
A-215	Mianus	700
A-217	Brighton	750
A-222	Redwood	700
A-305	Round Hill	350

account-number	branch-name	balance
A-101	Downtown	500
A-215	Mianus	700
A-102	Perryridge	400
A-305	Round Hill	350
A-201	Brighton	900
A-222	Redwood	700
A-217	Brighton	750

- No significance to ordering
- because both tables represent sets

Relational Database Schema

2.6

JCJ

- A database consists of one or more relations
- Information about an enterprise is broken down into its components
- Each relation stores just part of the information

E.g.: account: information about accounts
depositor: information about which customer
owns which account
customer: information about customers

- Storing all information as a single relation such as bank(account-number, balance, customer-name, ..) results in
 - repetition of information (e.g. two customers own an account)
 - the need for null values (e.g. represent a customer without an account)

BMS 2022 JCJ DBMS 2022 JCJ 2.8

Relational Database Schema (Cont.)

A relational database schema is a set of relation schemas, each with a distinct name.

E.g.: Relational database schema { account, customer}

account(Account No: Digits(10), Balance : Digits(8)) and

customer (Customer-name: Char (60),

Account No : Digits(10)) are the two relations in

the database schema...

Kevs

- Retrieval of data from a relation requires keys
- There are dierent types of keys:
- Super key
- Candidate key
- Primary key and
- Foreign key

DBMS 2022 2.9

Super keys

- A set of attributes K, whose values uniquely identify a tuple in any instance.
- If K is a super key any super set of k is also a super key.

customer

customer-ID	customer-name	customer-street	Category
5700893	Jones	Main	A
5700379	Smith	NULL	A
5700321	Curry	North	C
5700990	Lindsay	NULL	B

{customer-ID},{customer-ID,category}, {customer-name,customer-street}, ---(if no two customers have the same name) (customer-name),

2.11

{customer-ID} etc. are superkeys

JCJ 2.10

Candidate Keys

- They are super keys for which no proper subset is a super key.
- i.e. K is a candidate key if:
 - K is a super key
 - K is minimal (contains no super keys)

{customer-name}, { customer-ID} are candidate keys for Customer

2.12

Primary Key

- A primary key is a candidate key chosen by the database designer to identify tuples uniquely within a relation
- Primary keys are often underlined in a relation schema
- Note that a primary key is a candidate key, and a candidate key is a super key
- Candidate keys that are not the primary key are called alternate keys
- e.g. Customer (<u>customer-ID</u>, customer-name, customerstreet, category)

DBMS 2022 JCJ

Relational Algebra

2.13

- A formal query language associated with the relational model. Gives a procedural method of specifying a retrieval query.
- Consist of functions that accept relation instances as arguments and return a relation instance as result
- Forms the core component of a relational query engine.
- SQL queries are internally translated into RA expressions.
- RA provides a framework for query optimization.
- There are five fundamental operations, two auxiliary operations, and a number of operations derived from them.

2.15

Foreign Keys

- Relation schema may contain attributes which refers to a candidate key attributes of some other relation provided both the attribute and the candidate key attributes are having the same domain. Such keys are called foreign keys.
- Example: Employee (Empld, EmpName, DepNo)

Department (DepId, DepName)

The foreign key attribute DepNo in Employee relation refers to the primary key attribute Depld in Department relation.

- There can be more than one foreign key in a relation schema.
- It is possible for a foreign key in a relation to refer to the primary key of the relation itself

Example: univEmployee (empNo , name, sex, salary, dept, reportsTo) reportsTo is a foreign key referring to empNo of the same relation.

Every employee in the university reports to some other employee for administrative purposes - except the *vice- chancellor*, of course!

DBMS 2022 JCJ

Relational Algebra Fundamental Operations

2.14

Let R and S be expressions resulting in relation instances

 \blacksquare Selection $\sigma_{e}(R)$ ------ choose tuples (rows)

Projection $\Pi_{\text{attrl ist}}$ (R) ------ choose attributes columns

■ Union R ∪ S ------ pool tuples together

■ Set difference R - S ------ tuples in R but not in S

Cartesian product R x S ----- all tuple combinations

2.16

MS 2022 DBMS 2022 JCJ

Relational Algebra Auxiliary Operations

These are used mainly for resolving naming conflicts and to break down complex operations to simpler ones

- Assignment R ←S -----set the value of R to that of S
- Rename PnewName(R) -----change attribute names

DBMS 2022

Formal Definition

2.17

- A basic expression in the relational algebra consists of either one of the following:
 - A relation in the database
 - A constant relation
- Let E₁ and E₂ be relational-algebra expressions; the following are all relational-algebra expressions:
 - $E_1 \cup E_2$
 - $E_1 E_2$
 - $E_1 \times E_2$
 - $\sigma_p(E_1)$, P is a predicate on attributes in E_1
 - $\Pi_s(E_1)$, S is a list consisting of some of the attributes in E_1
 - $\rho_{x}(E_{1})$, x is the new name for the result of E_{1}

Relational Algebra Other Operations

- Set intersection R ∩ S ------ tuples in both R and S
- Conditional join R ⋈_θ S ------ selection on R x S
- Equijoin --conditional join with only equality comparison in predicate
- Natural join R ⋈ S ----- Equijoin on common attributes
- Division R÷S ------ "inverse" of R x S

DBMS 2022 JCJ
2.18

Selection

- A unary operation : $\sigma_{\theta}(R)$
- Selects tuples (rows) from a relation instance
- Resultant relation instance has the same schema as R
- It contains only tuples of R that satisfies the predicate θ
- The predicate is a Boolean expression. It consists of a combination of terms are comparison between two attribute values or between an attribute value and a constant.
- Allowed comparison are <,≤ , >, ≥ , =, ≠ Terms are connected by logical and (^) or logical or (V) operators

Selection – Example

Relation R

Α	В	С	D
α	α	1	7
α	β	5	7
β	β	12	3
β	β	23	10

• $\sigma_{A=B \land D > 5}(R)$

Α	В	С	D
α	α	1	7
β	β	23	10

DBMS 2022

2.21

Projection-Example

■ Relation R:

 $\blacksquare \Pi_{A,C}(R)$

$$\begin{array}{c|cccc}
A & C \\
\hline
\alpha & 1 \\
\alpha & 1 \\
\beta & 1 \\
\beta & 2 \\
\hline
\end{array}$$

$$\begin{array}{c|cccc}
A & C \\
\hline
\alpha & 1 \\
\beta & 1 \\
\beta & 2 \\
\hline$$

Projection

- A unary operation: π_{attrList} (R) Project attributes (columns) from a relation instance
- Resultant relation schema consist of attributes in attrList in the specified order
- Each tuple in the answer comes from a tuple in R, with only the specified attributes retained
- Duplicate tuples are eliminated

JCJ DBMS 2022

2.22

JCJ

JCJ

Union

- A binary operation: R ∪ S
- Resultant relation instance has schema the same as that of R (or S)
- R∪S is the set union of all the tuples that occur in either, or both, relation instances R or S
- R and S must be (union-)compatible. Compatible relations must satisfy the following criteria: they have the same number of attributes corresponding attributes, taken in order from left to right, have the same domains
- There are no duplicate tuples in the resultant relation instance

Union – Example

Relations R, S:

Α	В	
α	1	
α	2	
β	1	
r		

 $\begin{array}{c|c}
A & B \\
\hline
\alpha & 2 \\
\beta & 3 \\
\end{array}$

 $R \cup S$:

Α	В
α	1
α	2
β	1
β	3

DBMS 2022

2.25

Set Difference – Example

Relations R, S:

 A
 B

 α
 2

 β
 3

R – S:



Set Difference

- A binary operation: R S
- Resultant relation instance has schema the same as that of R (or S)
- The result contain tuples that occur in R but not in S.
- R and S must be compatible

DBMS 2022

2.26

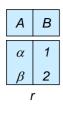
JCJ

Cartesian product

- A binary operation: R x S
- Also called cross-product and Similar to Cartesian product of sets
- Resultant relation instance has schema that contains, in order, all the attributes in R and then all the attributes in S. For every tuple $r \in R$ and $s \in S$, the concatenation of r and s is a result tuple of $R \times S$
- Assume that attributes of r(R) and s(S) are disjoint. (That is, $R \cap S = \emptyset$).
- If attributes of *r*(*R*) and *s*(*S*) are not disjoint, then renaming must be used.

Cartesian-Product -Example

Relations r, s:



С	D	Ε			
$\begin{array}{c} \alpha \\ \beta \\ \beta \\ \gamma \end{array}$	10 10 20 10	a a b b			

rxs:

Α	В	С	D	Ε
α	1	α	10	а
α	1	β	10	а
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	а
β	2	β	10	а
β	2 2 2	β	20	b
β	2	γ	10	b

DBMS 2022

2.29

Rename

- Gives relations or attributes a different name.
- Allows us to name, and therefore to refer to, the results of relationalalgebra expressions.
- Allows us to refer to a relation by more than one name. Rename gives relations or attributes a different name
- Unary operation
- Has a number of variations
- ρ_X(R) ----- rename the relation instance R as X
- $\rho_{X(A1,A2,...An)}(R)$ ------ rename the relation instance R with degree n as X, and the r-th attribute as Ar for r ϵ { 1,... n }
- $\rho_{(X_1 \to Y_1, X_2 \to Y_2, ..., X_m \to Y_m)}(R)$ ----- rename attribute name x_r as y_r in the relation instance R for $r \in \{1, 2, ... m\}$
- $\rho_{X_{(X_1 \to Y_1, X_2 \to Y_2, ..., X_m \to Y_m)}}(R)$ rename attribute name x_r as y_r for $r \in \{1, 2, ... m\}$ in the relation instance R renamed as X

Assignment

- Binary operation: R ←S
- Content of R is replaced by the content of S
- R and S must be compatible
- Resultant relation instance has the schema of R

Example: $T \leftarrow R \times S$:

!

Additional Operations

2.30

JCJ

We define additional operations that do not add any power to the relational algebra, but that simplify common queries.

- Set intersection
- Join Operations
- Division
- Assignment

DBMS 2022 JCJ

2.31

2.32

Set intersection

- A binary operation: R ∩ S
- The result contain tuples that occur both in R and S
- Resultant relation instance has schema the same as that of R (or S)
- R and S must be compatible
- $\blacksquare R \cap S = R (R S)$

Set-Intersection - Example

Relation R ,S: $\begin{bmatrix} A & B \\ \alpha & 1 \\ \alpha & 2 \\ \beta & 1 \end{bmatrix}$

A B α 2 β 3

S

R

 \blacksquare R \cap S A B

DBMS 2022 JCJ

DBMS 2022

2.33

Join operations

- Combine information from two or more relations
- Selections and projections of cross product of two relation instances
- A number of variants:
 - Conditional join (θ-join)
 - Equijoin
 - Natural join

Conditional join (θ-join)

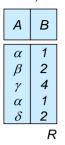
2.34

- R N S where R and S are relations and is θ is a predicate on the attributes of R together with that of S
- The predicate often refer to attributes of both R and S
- \blacksquare R \bowtie_{θ} S = σ_{θ} (R X S)

DBMS 2022 JCJ DBMS 2022 JCJ 2.35

θ- Join Example

Relations R, S:



С	D	Ε
1	а	α
3	а	β
1	а	γ
2	b	δ
3	b	\in
	S	

 $R\bowtie_{B>C} S$

Α	В	С	D	Ε
γ	4	1	а	α
γ	4	3	а	β
γ	4	1	а	γ
γ	4	2	b	δ
γ	4	3	b	€

DBMS 2022

2.37

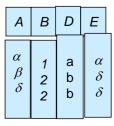
Equi Join Example

■ Relations R, S:

Α	В
$\begin{array}{c} \alpha \\ \beta \\ \gamma \\ \delta \end{array}$	1 2 4 2
	R

 $\begin{array}{c|cccc}
C & D & E \\
\hline
3 & a & \beta \\
1 & a & \gamma \\
2 & b & \delta \\
3 & b & \epsilon
\end{array}$

 $R\bowtie_{B=C} S$



Equijoin

- Equijoin is a special case of conditional join
- Predicate of a conditional join may contain only conjunction (logical AND) of equalities between two attributes in the two relations

M

- For R R.a=S.b S, it is redundant to include both R.a and S.b in the result
- A projection is done so that only the R.a is retained
- Join operation with this refinement is called equijoin

DBMS 2022

JCJ

2.38

JCJ

Natural-Join

- Notation: R X S
- Equality on attributes with the same name
- Equijoin in which equalities are specified on all fields having the same name in the operands
- The result will not have two fields with the same name

Natural Join - Example

Relations R, S:

Α	В	С	D
α	1	α	а
β	2	γ	а
γ	4	β	b
α	1	γ	а
δ	2	β	b
r			

В	D	Ε
1 3 1 2 3	a a a b b	α β γ δ \in
S		

 $R\bowtie S$

Α	В	С	D	Ε
α	1	α	а	α
α	1	α	а	γ
α	1	γ	а	α
α	1	γ	а	γ
δ	2	β	b	δ

DBMS 2022 2.41

Outer Joins (Cont.)

- Left outer join: It keeps all tuples in the first, or left relation R in the result. For some tuple *t* in R, if no matching tuple is found in *s* then S- attributes of *t* are made null in the result.
- Right outer join: Same as above but tuples in the second relation are all kept in the result. If necessary, R- attributes are made null.
- Full outer join: All the tuples in both the relations *r* and *s* are in the result.

Outer Joins

- Theta join,equi-join and natural join are all called *inner joins*. The result of these operations contain only the matching tuples.
- The set of operations called outer joins are used when all tuples in relation R or relation S or both in R and S have to be in result.
- There are 3 kinds of outer joins:
 - Left outer join R S
 - Right outer join R 🖂 S
 - Full outer join R ⊃ S

DBMS 2022 JCJ 2.42

Example

■ Relation loan

loan-number	branch-name	amount
L-170	Downtown	3000
L-230	Redwood	4000
L-260	Perryridge	1700

Relation borrower

customer-name	loan-number
Jones	L-170
Smith	L-230
Hayes	L-155

Inner Join

Inner Join

loan ⋈ Borrower

loan-number	branch-name	amount	customer-name
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith

loan-number	branch-name	amount
L-170	Downtown	3000
L-230	Redwood	4000
L-260	Perryridge	1700

customer-name	loan-number
Jones	L-170
Smith	L-230
Hayes	L-155

Left Outer Join

■ Left Outer Join

loan-number	branch-name	amount	customer-name
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith
L-260	Perryridge	1700	null

loan-number	branch-name	amount
L-170	Downtown	3000
L-230	Redwood	4000
L-260	Perryridge	1700

customer-name	loan-number
Jones	L-170
Smith	L-230
Hayes	L-155

JCJ

DBMS 2022

2.45

DBMS 202

JCJ

2.46

Outer Join – Example

■ Right Outer Join

loan ⋈ borrower

loan-number	branch-name	amount	customer-name
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith
L-155	null	null	Hayes

■ Full Outer Join

loan

□ borrower

loan-number	branch-name	amount	customer-name
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith
L-260	Perryridge	1700	null
L-155	null	null	Hayes

Division Operation

- Binary operation: R ÷ S, or R / S
- Resultant relation schema is that of R without attributes in S (i.e R – S)
- A tuple will appear in the result if every combination tuples in S and itself appears in R
- Suited to queries that include the phrase "for all".
- $R(A_1, ..., A_m, B_1, ..., B_n)$ $S(B_1, ..., B_n)$

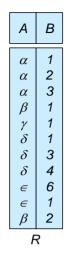
The result of R ÷ S is a relation on schema

$$R - S(A_1, ..., A_m)$$

 $R \div S =$

Division Operation – Example

Relations R. S:



В

2

R ÷ S:

DBMS 2022



JCJ

Why "division"

Integer division: let Q = A / B, where Q, A, and B are all

Relational algebra division: let $Q = A \div B$, where Q, A, and B are all relation instances. Loosely, Q is the largest relation instance such that $Q \times B \subset A$

integers. Q is the greatest integer such that Q x B <= A

Practical systems often do not implement division

Another Division Example

Relations R. S:

Α	В	С	D	Ε
α	а	α	а	1
α	а		а	1
α	а	γ	b	1
β	а	γ	а	1
$\begin{array}{ccc} \alpha & & \\ \alpha & & \\ \alpha & & \\ \beta & & \\ \beta & & \\ \gamma & & \\ \gamma & & \\ \gamma & & \\ \end{array}$	а	γ γ γ γ	b	3
γ	а	γ	а	1
γ	а	γ γ	b	1
γ	а	β	b	1
R				

Ε а b S

JCJ

R ÷ S:

Α	В	С
α	а	γ
	а	27

2.50

Banking Example

branch (branch-name, branch-city, assets)

customer (customer-name, customer-street, customer-only)

account (account-number, branch-name, balance)

loan (loan-number, branch-name, amount)

depositor (customer-name, account-number)

borrower (customer-name, loan-number)

Example Queries

Find all loans of over Rs.1200

■Find the loan number for each loan of an amount greater than Rs.1200

$$\prod_{loan-number} (\sigma_{amount > 1200} (loan))$$

Example Queries

■ Find the names of all customers who have a loan, an account, or both, from the bank

$$\Pi_{customer-name}$$
 (borrower) $\cup \Pi_{customer-name}$ (depositor)

Find the names of all customers who have a loan and an account at bank.

$$\Pi_{customer-name}$$
 (borrower) $\cap \Pi_{customer-name}$ (depositor)

DBMS 2022 JCJ 2.53

Example Queries

■ Find the names of all customers who have a loan at the Idukki branch.

$$\Pi_{\textit{customer-name}} (\sigma_{\textit{branch-name}="ldukki"} \\ (\sigma_{\textit{borrower.loan-number}} (\textit{borrower x loan})))$$

Find the names of all customers who have a loan at the ldikki branch but do not have an account at any branch of the bank.

$$\Pi_{customer-name}$$
 ($\sigma_{branch-name}$ = "Idukki"

($\sigma_{borrower.loan-number}$ = loan.loan-number(borrower x loan))) - $\Pi_{customer-name}$ (depositor)

DBMS 2022 JCJ

Example Queries

2.54

Find the names of all customers who have a loan at the Idukki branch.

```
-Query 1  \prod_{\text{customer-name}} (\sigma_{\text{branch-name}} = \text{``Idukki''} (\sigma_{\text{borrower.loan-number}} = \text{loan.loan-number} (\text{borrower x loan})))
```

$$\begin{split} &- \text{ Query 2} \\ &\prod_{\text{customer-name}} (\sigma_{\text{loan.loan-number}} = \text{borrower.loan-number} (\sigma_{\text{branch-name}} = \text{`'ldukki''}(\text{loan})) \text{ x borrower})) \end{split}$$

MS 2022 JCJ DBMS 2022 JCJ 2.55

Example Queries

Find the largest account balance

- Rename account relation as d
- The query is:

```
\Pi_{balance}(account) - \Pi_{account.balance} (\sigma_{account.balance} < d.balance (account x <math>\rho_d (account)))
```

Example Queries

■ Find all customers who have an account from at least the "Thodupuzha" and the Idukki" branches.

Query 1

$$\Pi_{\text{CN}}(\sigma_{\textit{BN}=\text{"Thodupuzha"}}(\textit{depositor} \mid \textit{account})) \cap \\ \Pi_{\text{CN}}(\sigma_{\textit{BN}=\text{"Idukki"}}(\textit{depositor} \mid \textit{account}))$$

where *CN* denotes customer-name and *BN* denotes branch-name.

DBMS 2022 JCJ 2.57

Example Queries

Find all customers who have an account at all branches located in Kochi city.

$$\prod_{customer-name, branch-name} (depositor \bowtie account)$$
 $\vdots \prod_{branch-name} (\sigma_{branch-city = \text{``Kochi''}} (branch))$

DBMS 2022 JCJ

2.58

DBMS 2022 2C.59