Date.....

Date.....

* Data

Facts that can be recorded and that have implicit meaning.

* Database

It is the collection of inter-related data which helps in efficient retrieval, insertion and deletion of data from database and organizes the data in the form of tables, views, schemas, reports etc. For ex- University database organizes the data about students, faculty and staff, administrator which helps in efficient retrieval, insertion and deletion of data from it.

→ DBMS

The software which is used to manage database is called DBMS. For ex - MY SQL, ORACLE are both popular commercial DBMS using different applications. DBMS allows users the following tasks -

(i) Data Definition.

It helps in creation, modification and removal of definition that defined in organization of data in database.

(ii) Data updation

Helps in insertion, modification and deletion of the actual data in the database.

Spiral

Spiral

Scanned by CamScanner

Date.....

(iii) Data Retrieval

Helps in retrieval of data from the database which can be used by application for various purposes.

(iv) User Administration

It help in ~~administering~~ registering and monitoring users. Enforcing data security, monitoring performance, maintaining data integrity, dealing with redundancy control, Being recovering info. corrupted by unexpected failure.

* Redundancy of data

In Data redundancy is a condition created within a database or data storage technology in which the same piece of data is held in two separate places.

* Inconsistency of data

Data inconsistency is a condition that occurs b/w files when similar data is kept in different formats in two different files, or when matching of data must be done b/w files. As a result of the data inconsistency, these files duplicate some data such as address and names, compromising data integrity

* Difficult data access

Spiral

Date.....

1. Unauthorized access
2. BACKUP OR RECOVERY
3. DATA ISOLATION
4. ATOMICITY PROBLEM

Spiral

Scanned by CamScanner

Date

Date

* Benefits of DBMS system

-) Reduction of the amount of data redudancy.
-) No more data inconsitency.
-) Store data can be shared by a single or multiple users.
-) Data integrity can be maintained it means you have only one data.
-) Security of data can be implemented.
-) Data indepedence can be achieved.

* RDBMS [Relational Data Base Management System]

-) It is a DBMS which is based on relational modules. as introduced by DOCTOR E.F.COTT.
-) RDBMS stores data in the form of related tables.

RDBMS	DBMS
-) It can be specify at a time of table creation.	-) Relationship blw two tables or files. are maintained by some program.
-) It supports client server architecture	-) Does not support client server architecture.
-) It supports distributed database.	-) Does not support distributed database.
-) Login at OS Level or command level	-) No security of data

Spiral

Spiral

Scanned by CamScanner

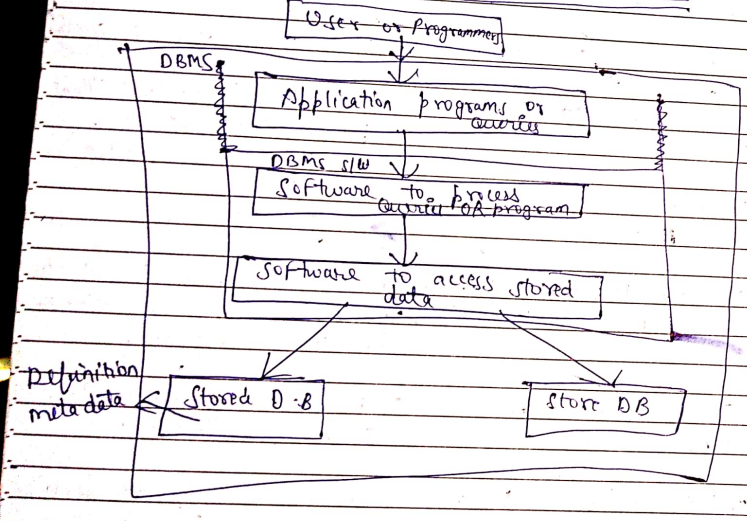
Meta data → data about data
stored data definition

Date

=) RDMS follows more than 7 or 8 rules of Cott

-) 7 to 8 rules of Cott follows

* Database Management System Environment



* DBMS Architecture and data independence

Spiral

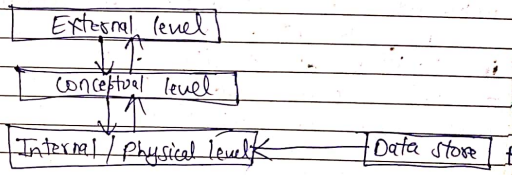
Date

In a three tier architecture we have :-

1. Internal level → (Internal schema)
2. Conceptual level (Conceptual schema)
3. External level (External schema)

Data model

A collection of concepts that can be used to describe the structure of a database.



(i) **Physical level**

It divides the complete system into three inter-related but independent modules-

(i) **Internal level / Physical level**

The information about location of data object in data store is kept. Various users of DBMS are unaware about the location of these objects.

(ii) **Conceptual level**

Data is represented in the forms of various data tables for ex - student database may contain student and course tables which will be visible to users. But users are unaware

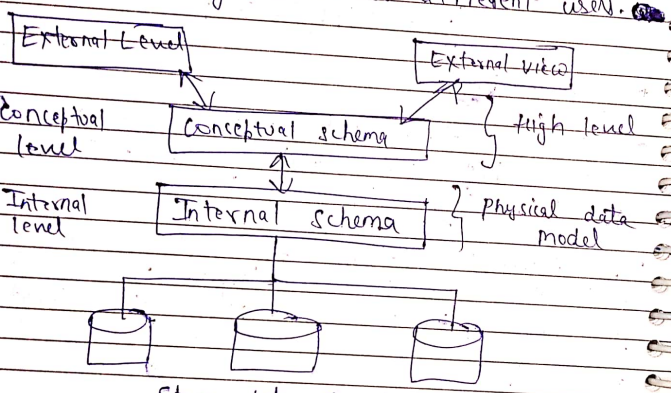
Spiral

Scanned by CamScanner

about their storage.

(iii) External level

It specifies the view of the data in terms of conceptual level tables. Each external level view is used to cater the needs of a particular category of users. For ex- Faculty of university is interested in looking course details of students. or students are interested in looking all details related to academics, accounts, courses and hostel details. So different views can be generated for different users.



* Data independence

The change of data at one level could not affect another level.

We have two types of data independence in

(i) Physical data independence

It means any change in physical location of tables and indexes should not effect conceptual level or external view of data.

(ii) Conceptual data independence or Logical data independence

This means change in conceptual schema (overall database) should not effect external schema. For ex- adding or deleting attributes of a table should not effect user's view of table.

* Data modelling using E-R models

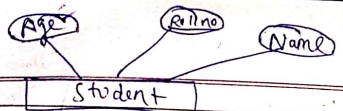
Entity

It is an object with a physical existence or a may be a conceptual existence.
physical - person, car
conceptual - job, company.
It is represented by rectangle.
A collection of entity is called entity set

Attributes

It define the property of entity.

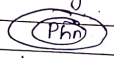
Scanned by CamScanner



Entity type

Date

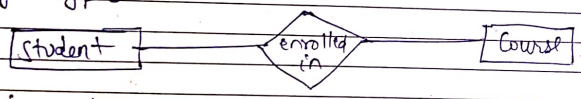
2. Multivalued Attribute – An attribute consisting more than one value for a given entity. For ex – phn no.



3. Derived Attribute – An attribute which can be derived from other entities of the entity type For ex - age

Relationship type and set

A relationship type represents the association b/w entity types.

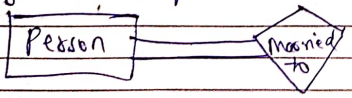


It is represented by diamond.

Degree of a relationship set

(i) Unary relationship

When there is only one entity set participating in a relation, the relationship is called as unary relationship.



Spiral

Date

2- Binary relationship

When there are two entity set participating in a relation. For ex – student will enroll in course.

3- n-ary relationship

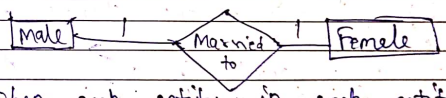
When we have n – relation participating in relationship.

Cardinality

The no. of times an entity of an entity set participates in a relationship set is known as cardinality.

It is of two types :

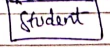
(i) One to one



When each entity in each entity set can take part only once in the relationship.

(ii) Many to one

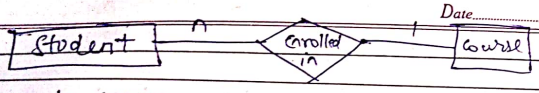
When entities in one entity set can take part only once in the relationship set and entities in other entity set can take part more than once in the relationship set. For ex - one student can take a single course like you. But that course can be taken by many.



Spiral

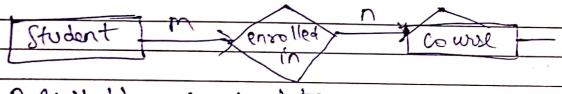
Scanned by CamScanner

Date



3- Many to many

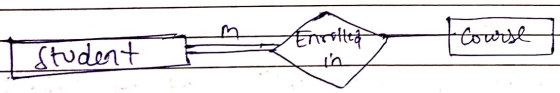
When entities in all entity sets can take part more than once in the relationship, the cardinality is many to many. For ex - if it is possible one student can take many courses and one course can be taken by many student



* Participation constraints

(i) Total participation

Each entity in the entity set must participate in the relationship. For ex - the students in a btech of a college.
If each student must enroll in a course, the participation of a student will be total and it will be defined by double line.



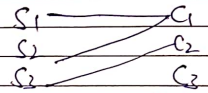
(ii) Partial participation

The entity in the entity set may or may not participate in the relationship.

Spiral

SHALOM

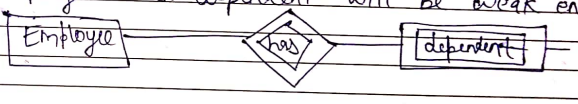
Date



A relationship set enrolled in where students are participating in total but course entity set is participating in partial.

Weak entity type

An entity type has a key attribute which uniquely identifies each entity in the entity set. But there exists some entity type for which key attributes cannot be defined. These are called weak entity type. For ex - a company may store the information of dependent such as parents, children, spouse. of an employee. but the dependence don't have existence without the employee. So dependent will be weak entity type



Spiral

Scanned by CamScanner

Date

Spiral

Date

* Instances and schema

Collection of information stored in the database at a particular moment is called an instance of the database. The overall design of a database is called database schema.

*DBMS keys

A key is a attribute or a combination of attribute that is used to identify records.

(i) Super key

An attribute or a combination of attribute that is used to identify the records uniquely is known as super-key. A table may have many super keys. Ex- Em_Id, NAME, ADDRESS,

(ii) Candidate key

It can be defined as minimal super key or irreducable of superkey. In other words an attribute or combination of attribute that identify the record uniquely but none of its proper subsets can identify the records uniquely.

(iii) Primary key

A candidate key that is used by the database designer for unique identification of each rec.

Spiral

Scanned by CamScanner

in a table is known as Primary key. A primary key can consist of one or more attribute of a table.

(iv) Foreign key

A foreign key is an attribute or a combination of attribute in one base table that points to the candidate key. (Generally is a primary key) of another table. Foreign key value must match an existing value in the parent table being NULL.

(v) Comparite key

If we use multiple attribute to create a primary key then that primary key is called comparite key.

(vi) Alternate key

It can be any of the candidate keys except for the primary key - Ex - Name and address

(vii) Secondary key

The attributes that are not even the superkey but can still be used for identification of records (not unique). Ex- Name, address, salary.

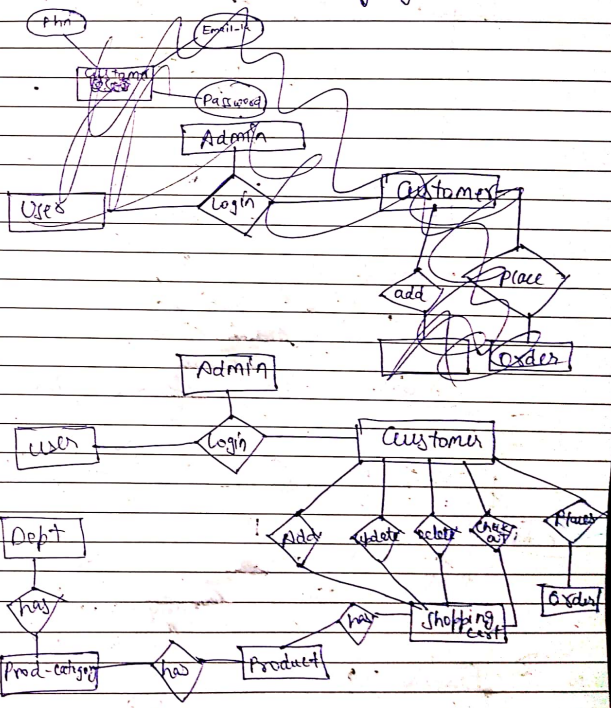
Spiral

Railway, Hotel management, hospital management, university Registration, Library management with a set of patient & Medical doctor

Date

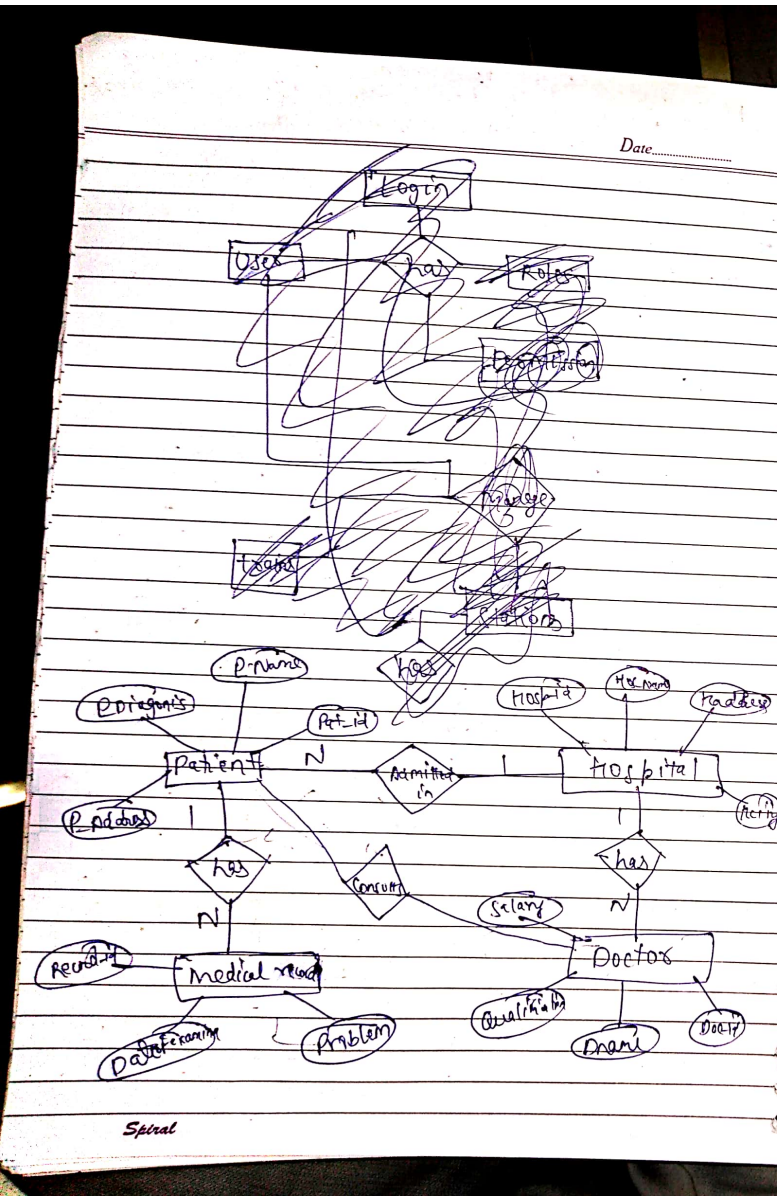
E-R Diagram of Food ordering system

Shopping cart, product, order, department, administrator, Product category, customer.



Spiral

Scanned by CamScanner



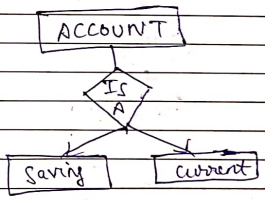
Date.....

* Enhanced ER model

(i) Generalisation
(ii) Specialisation
(iii) Aggregation

(i) Generalisation

It is a bottom up approach in which two lower level entities combined to form higher level entities. ~~It is a process of defining~~ Entities are combined to form a more generalised entity. In other words subclasses are combined to form a superclass.

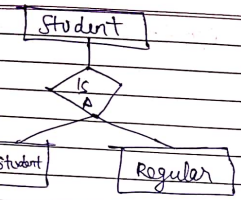


(ii) Specialisation

It is a process of defining a set of subclass of an entity type, this entity type is called the superclass of this specialisation. It is opp. to generalisation and follows top down approach in which one higher level entity can be broken down into two lower level entity.

Scanned by CamScanner

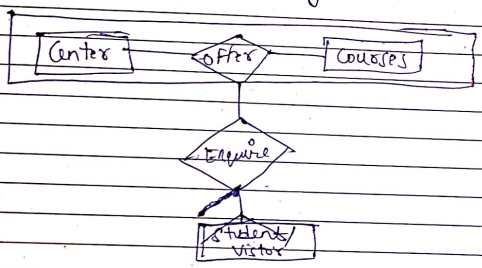
Date.....



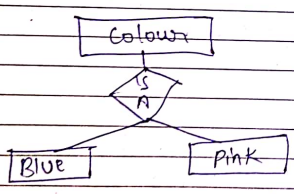
Top down approach

(iii) Aggregation

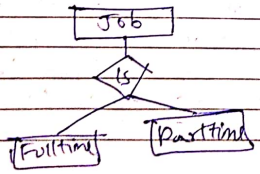
It is the process when relation b/w two entities treated as a single entity.



(i)



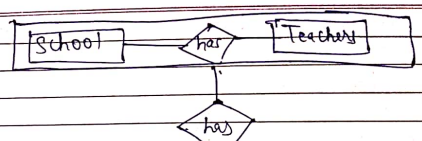
(ii)

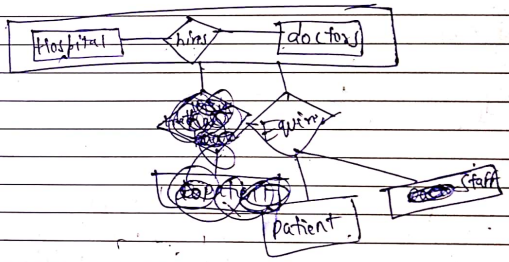


Spiral

Date.....

(iii)





* Relational Model

It represents the database as a collection of relations which is nothing but a table of values. Every row in the table represents a collection of related data values. These rows in the table denote a real world entity or relationship.

* Relational Model Concepts

1. Attribute — column in a table.

Attributes are the properties which define a relation.

Spiral

Scanned by CamScanner

Date

2- Tables :- A relation are saved in the table format. It has two properties one is row and other is column. Rows represent record and column represents attributes. A single row of a table having a single record is called tuple.

3- Relation schema

It represents the name of the relation with its attributes.

4- Degree

The total no. of attributes (column) which in the relation exist is called the degree of relation.

5- Cardinality

The total no. of rows in a table.

* Relational Integrity Constraints

It is a condition that can be applied on the database schema to restrict the data according to the need. The condition is satisfied then only it can be stored in the database. It is to ensure that there should not be any loss in data consistency due to changes made to the database by the authorised users.
Like - Phone no. (are always 10 digits)

Spiral

Date

We have different type of constraints.
1- Entity - Integrity constraint
2- Referential - Integrity constraint
3- Domain constraint
4- Key constraint

A relation schema R
(i) denoted by $R(A_1, A_2, \ldots A_n)$ is made up of relation name R and a list of attributes $A_1, A_2 \ldots A_n$.
(ii) Each attribute A_i is a name of role played by some domain D in the relation schema are there called a Domain of A_i. and it denoted by $dom(A_i)$.

- The data types describing the types of values that can appear in each column is called domain.
- A Domain D is a set of atomic values By atomic we mean that each value in the domain is indivisible as far as the relational model is concerned.

For ex- Phone no, Aadhar no.,

1. Entity - Integrity Constraint

It states that no primary key value can be NULL.

2. Referential Integrity Constraint

It ensures that a value that appears in one relation for a given set of attributes also

Spiral

Scanned by CamScanner

Date

appear for a certain set of attribute in another relation. It is the foreign key. For ex.

Customer

Cust ID	Name	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive

Billing

Invoice no	Cust Id	Amount
1	1	100
2	1	200
3	2	300

The total amount associated with google is 300 Here Cust_Id acts as a Foreign key.

3. Domain Constraint

Domain is a set of atomic values which means that each value in the domain is individual as far as the relation model is concerned. It specifies that the value of each attribute A must be an atomic value from the domain dom(A).

For ex.

Spiral

Date

→ Aadhaar no. (Set of 12 digits) or (Set of valid 12 digits in India) India mobile no.

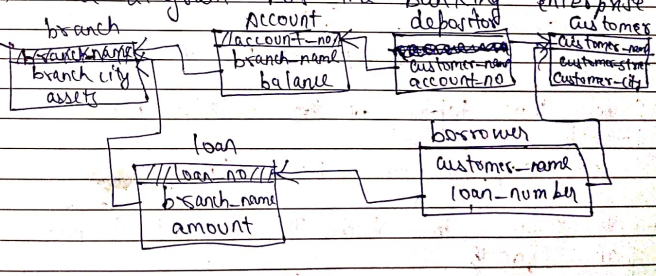
→ Employee age (possible ages of a employee of a company each must be a value btw 22 and 60 yrs old)

4. Key Constraint

A relation is defined as a set of tuples. All tuples in a relation must be distinct. This means that no two tuples can have the same combination of values for all their attributes (columns).

* Relational Algebra

→ Schema diagram for the banking enterprise



Spiral

Scanned by CamScanner

Date

- Relational Algebra is a procedural query language. It consist of set of operations that take one or two relation as input and produce the new relation as their result.
- The fundamental operations are -
  - (i) Select
  - (ii) Project
  - (iii) Union
  - (iv) Set - difference
  - (v) Cartesian product
  - (vi) Set - intersection
  - (vii) Natural joint
  - (viii) Division
  - (ix) Assignment

(i) Select - Select tuples that satisfy the given predicate. The selection in a relational algebra is denoted by σ.

$\sigma_{predicate}(\_\_\_\_)$ ← Argument Relation

→ Select those tuples of loan relation where the branch is Noida

$\sigma_{branch\_name = "Noida"}(loan)$

→ All tuples in which the amount is more than 1200 $

$\sigma_{amount > 1200}(loan)$

Comparision = equals to, ≥ greater than equal to, ≠ not equal to

Spiral

Date

We can combine several predicates into a larger predicates by using connectives

And - ∧ OR → ∨ Not → ¬

→ Find all tuples pertaining to loans of more than 1200 $ made by the Noida branch

Ans

$\sigma_{branchname = "Noida"(loan) \wedge amount > 1200(loan)}$

(ii) Project

It is used for columns. It is a unary operation that returns its argument relation, with certain attributes left out. denoted by Π.

$\Pi_{attribute}(Argument\ Relation)$

→ It is list all loanno. and the amount of the loan

$\Pi_{loanno, amount}(loan)$

Composition type →

(i) Find those customer name who lives in Noida.

$\Pi_{customer\_name}(\sigma_{customercity = "Noida"}(Customer))$

Spiral

Scanned by CamScanner

(iii) Union

Date

(ii) Find the names of all bank customers who have either an account or an loan or both.

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

(iii) Set-difference (−)

It allows us to find tuples that are in one relation but are not in another.

→ Find out the customer with an account but no loan.

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

(iv) Cartesian product operation (×)

It allows us to combine information from any two relations.

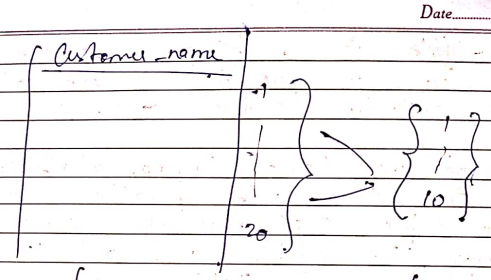
→ R_1 → borrower
R_2 → loan
$R_1 \times R_2$ = borrower × loan = R

Forex - Find the names of all customers who have a loan at noida branch.

Spiral

Date

Customer_name



$\Pi_{customer\_name}(\sigma_{loan.loan\_no = borrower.loan\_no}(\sigma_{branch\_name = "Noida"}(loan \times borrower)))$

(vi) Set-intersection operation (∩)

Find out customers who have a loan and an account. (both)

$\Pi_{customer\_name}(borrower) \cap \Pi_{customer\_name}(depositor)$

(vii) Natural join operation (⋈)

It performs a selection forcing equality on those attributes that appear in two relation schema and finally removes duplicate attributes.

→ Find the name of all customers who have a loan at the bank alongwith the loan no and the loan amount.

$\Pi_{customer\_name, loanno, amount}(\sigma_{loan.loanno = borrower.loanno}(loan \times borrower))$

$\Pi_{customer\_name, loanno, amount}(borrower \bowtie loan)$

Spiral

Scanned by CamScanner

- Natural join is a binary operation that allows us to combine certain selection and a cartesian product into one operation.

Borrower and loan have loanno. attribute, in both, the natural join operation considers only pair of tuples that have the same value on loanno. It combine each such pair of tuples into a single tuple on the union of two schemas.

-) Find the names of all branches with customers who have an account in the bank and who live in Noida.

~~branchname~~ = "Noida" (~~Account~~ depositor)

$\Pi_{branchname}(\sigma_{customercity = "Noida"}(account \bowtie depositor \bowtie customer))$

-) Find all customers who have both a loan and a account at the bank.

$\Pi_{customer\_name}(borrower \bowtie depositor)$

(viii) Division operator (÷)

It is suited to queries that include the phrase "for all". The result consist of the restriction of the tuple in R to the attribute name

Spiral

unique to R i.e. in the header of R but not in the header of S for which it holds that all their combination with tuple in S are present are present in R

Completed ÷ DB Project

Student	Task
X	Db 1
X	2
X	Comp 1
Y	Db 1
Y	Comp 1
Z	Db 1
Z	Db 2

Task
Db 1
Db 2

Student
X
Z

Q Use a attribute are id, age, gender, occupation_id or city_id

Occupation: occ_id, occupation_name

city : city_id, city_name

Table 1-

Id	Name	Age	Gender	Occ Id	City Id
1	John	28	Male	1	2
2	Sara	30	Female	3	4
3	Victor	31	Male	2	5
4	Jaina	27	Female	1	3

Table 2 : occupation

ID	Name
1	Software engineer
2	Accountant
3	Doctor
4	Library assistant

Spiral

Scanned by CamScanner

Table 3 : City

id	Name
1	Helifax
2	Calgary
3	Boston
4	New yosk
5	Toronto

* Solve the following relational expression for above relation.

(i) Pname (Rage > 25 (user))
(ii) User ⋈ occupation ⋈ city
(iii) Rusr_occ.id = occupation.occupation ID (user X occupation)
(iv) Pname, gender (Rcityname = "Boston" (user ⋈ city))

ci) Sara, Victor, Jaine
(ii)
(iii) Jaine, Female
(iv)

4	Jaine	27	Female	1	3 Boston

Q Consider the DB withe following schema -

1. Person (Name, age, gender)
2. Frequents (Name, pizzeria)
3. Eats (Name, Pizza)
4. Serves (Pizzeria, Pizza, price)

Queries : ① Find all pizzerias frequented by atleast 1 person under the age of 18.
② Find the name of all females who eat either mushroom or plain pizza or both.

Spiral

Date

① Find all pizzerias that serves at least 1 pizza that anny eats for less than 10 $

Ans 1 - Πpizzeria (σage < 18 (Person ⋈ Frequent))
2. Πname (σgender = 'F' (σpizza = "Mushroom" () σpizza = "Plane Pizza (Persons ⋈ Eats)))
3. Πpizzeria (σname = "Anny" (σprice < 10 (Person ⋈ eats ⋈ serves)))

* Tuple Relational Calculus

- A nonprocedural query language, where each query is of the form
  {t | P(t)}
- It is the set of all tuples t such that predicate P is true for t
- t is a tuple variable, t[A] denotes the value of t on attribute A
- t ∈ r denotes that tuple t is in relation r
- P is a formula similar to that of the predicate calculus.

* Predicate calculus formula

- Set of attributes and constants
- Set of comparison operators (eg <, ≤, =, ≠, >, ≥)
- Set of connectives : and (∧), or (∨), not (¬)
- Implication (⇒) : x ⇒ y, if x is true then y is true x ⇒ y = ¬x ∨ y

Spiral

Scanned by CamScanner

Date.....

→ set of quantifiers :-

→ $\exists t \in r (Q(t))$ = "there exists" a tuple in t in relation r such that predicate Q(t) is true

→ $\forall t \in r (Q(t))$ = Q is true "for all" tuples t in relation r.

Ex Find the ID, name, dept_name, salary for instructors whose salary is greater than $80000

Ans $\{t \mid t \in$ instructor $\wedge\ t[salary] > 80000\}$

Notice that a relation on schema (ID, name, dept_name, salary) is implicitly defined by the query.

→ As in the previous query, but output only the ID attribute value

$\{t \mid \exists s \in$ instructor $(t[ID] = s[ID] \wedge s[salary] > 80000)\}$

Notice that a relation on schema (ID) is implicitly defined by the query.

Ex Find the name of all instructors whose dept. is in the Watson building

$\{t \mid \exists s \in$ instructor $(t[name] = s[name])$
$\wedge \exists u \in$ department $(u[department\ name] = s[department\ name]$
$\wedge\ u[building] =$ "Watson")$\}$

Spiral

Date.....

* Find all employees whose salary is above 50000

Ans $\{t \mid t \in$ employee $\wedge\ t[salary] > 50000\}$

* Find the branch name, loan no. and amount for loans of over 1200

Ans $\{t \mid t \in$ loan $\wedge\ t[amount] > 1200\}$

Query 1. Find the loan no. for each loan of an amount greater than 1200

2. Find the name of all customers who have a loan from Noida branch.

Ans 1. $\{t \mid \exists s \in$ loan $(t[loan\text{-}no.] = s[loan\ no] \wedge s[amount] > 1200)\}$

The set of all tuples t such that there exist a tuple s in relation loan for which the values of t and s for the loan no. attribute are equal & the value of s for the amount attribute is greater than 1200

2. $\{t \mid \exists s \in$ borrower $(t[customer\_name] = s[customer\_name]$
$\wedge \exists u \in$ loan $(u[branch\_name] =$ "Noida" $\wedge$
$u[loan\text{-}no.] = s[loan\text{-}no.]))\}$

* Domain Relational Calculus

Customer

cust_name	city	Street
A	Noida	P
B	Calcutta	Q
C	Bijnor	R
D	Spiral Bangalore	S

loan

Loan no.	Branch name	Amount
L01	Main	200
L03	Main	150
L10	Sub	90
L08	Main	60

Borrower

Cust name	Loan no.
D	L01
A	L08
C	L03

Scanned by CamScanner

Date

It is a non-procedural query language equivalent in power to tuples relational calculus. It provides only the description of the query but it doesn't provide the method to solve it. A query is expressed as

$\{\langle n_1, n_2, n_3, \ldots n_n \rangle \mid P(n_1, n_2, n_3 \ldots n_n)\}$

where $\langle n_1, n_2, n_3 \ldots n_n \rangle$ represents resulting domain variables $P(n_1, n_2, n_3 \ldots n_n)$ represent the condition of formula equivalent to predicate calculus

Query: Find the loan no, branch, amount of loans of greater than or equal to 100 amount.

$\{\langle l, b, a \rangle \mid \langle l, b, a \rangle \in loan \wedge (a \geq 100)\}$

2. Find the loanno. for each loan of an amount greater or equal to 150

$\{\langle l \rangle \mid \langle l \rangle \in loan \wedge (a \geq 150)\}$

3- Find the name of all customers having a loan at the main branch and find the loan amount.

$\{\langle c, a \rangle \mid \exists l(\langle c, l \rangle \in borrower \wedge \exists b(\langle l, b, a \rangle \in loan \wedge b = \text{"main"}))\}$

Spiral

Date

Spiral

Scanned by CamScanner