

Database Management (4331603) - Winter 2024 Solution

Milav Dabgar

December 07, 2024

પ્રશ્ન 1(a) [3 ગુણ]

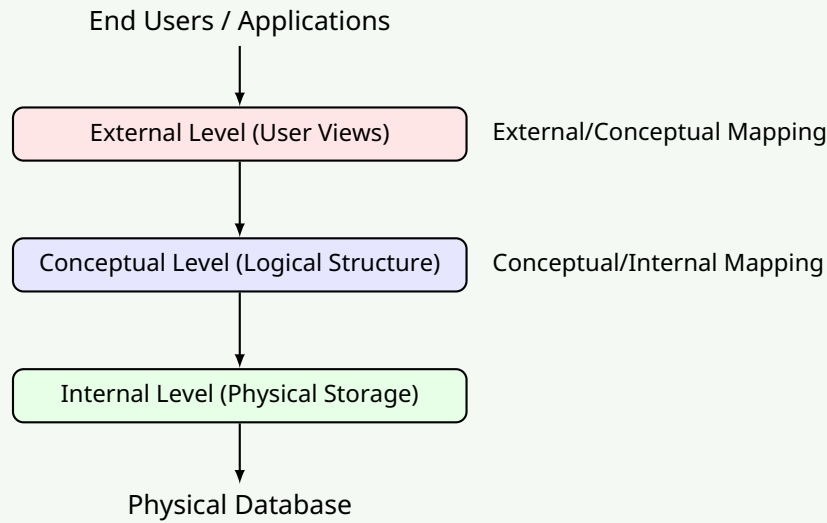
Three-level database architecture સમજાવો.

જવાબ

જવાબ:

કોષ્ટક 1. Three Levels of Architecture

| Level | Description | Purpose |
|------------------|-------------------------------------|-----------------------------|
| External Level | User views અને application programs | Data abstraction for users |
| Conceptual Level | Complete logical structure | Organization-wide data view |
| Internal Level | Physical storage details | Storage અને access methods |



આકૃતિ 1. Three-Level Database Architecture

- **External Level:** Individual user views અને specific application requirements દર્શાવે છે.
- **Conceptual Level:** Storage details વગર complete database schema દર્શાવે છે.
- **Internal Level:** Physical storage structures અને access paths દર્શાવે છે.

મેમરી ટ્રીક

“ECI - Every Computer Interface”

પ્રશ્ન 1(b) [4 ગુણ]

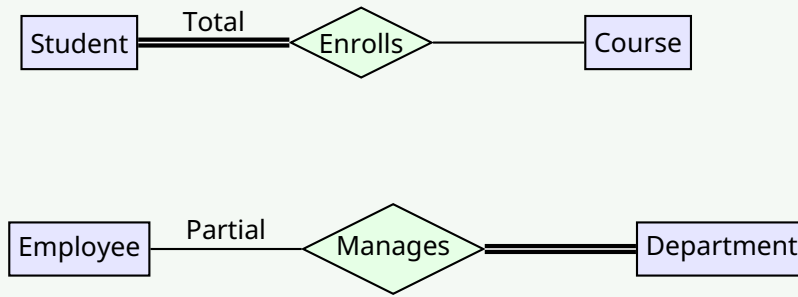
Total Participation અને Partial Participation ઉદાહરણ સાથે સમજાવો.

જવાબ

Table:

કોષ્ટક 2. Participation Types

| Participation Type | Definition | Symbol | Example |
|-----------------------|---------------------------------------|-------------|--------------------------------|
| Total Participation | દરેક entity એ participate કરવું જ પડે | Double line | Student-Course enrollment |
| Partial Participation | અમુક entities participate ન પણ કરે | Single line | Employee-Department management |



આકૃતિ 2. Participation Constraints

- **Total Participation:** બધા Students એ ઓછામાં ઓછા એક Course માં enroll થવું ફરજિયાત છે.
- **Partial Participation:** બધા Employees કઈ department manage કરતા નથી.
- **Double lines** total participation constraints દર્શાવે છે.
- **Single lines** partial participation relationships દર્શાવે છે.

મેમરી ટ્રીક

“Total = Two lines, Partial = Plain line”

પ્રશ્ન 1(c) [7 ગુણ]

File management systems ની સરખામણીમાં DBMS ના ફાયદા સમજાવો.

જવાબ

Table:

કોષ્ટક 3. DBMS Advantages

| Advantage | File System | DBMS |
|--------------------|---------------------|-----------------------------|
| Data Redundancy | High duplication | Controlled redundancy |
| Data Inconsistency | સામાન્ય સમસ્યા | Data integrity જળવાઈ રહે છે |
| Data Sharing | Limited sharing | Concurrent access support |
| Security | File-level security | User-level access control |
| Backup & Recovery | Manual process | Automatic mechanisms |

- **Reduced Data Redundancy:** Applications વચ્ચે duplicate data storage દૂર કરે છે.

- **Data Consistency:** બધી applications માં uniform data સુનિશ્ચિત કરે છે.
- **Data Independence:** Applications ને data structure changes થી સ્વતંત્ર રાખે છે.
- **Concurrent Access:** એકથી વધુ users data ને એકસાથે access કરી શકે છે.
- **Security Control:** User authentication અને authorization mechanisms પૂરા પાડે છે.
- **Backup and Recovery:** Automatic data protection અને restoration પૂરું પાડે છે.
- **Data Integrity:** Constraints દ્વારા data quality જાળવી રાખે છે.

મેમરી ટ્રીક

“RDCCSBI - Really Don't Copy, Control, Secure, Backup, Integrate”

પ્રશ્ન 1(c) OR [7 ગુણ]

વિવિધ data models ની યાદી બનાવો. કોઈપણ બે સંક્ષિપ્તમાં સમજાવો.

જવાબ

Data Models List:

- Hierarchical Data Model
- Network Data Model
- Relational Data Model
- Object-Oriented Data Model
- Entity-Relationship Model

કોષ્ટક 4. Comparison of Data Models

| Model | Structure | Advantages | Disadvantages |
|------------|--------------------------|----------------------|----------------------|
| Relational | Tables with rows/columns | Simple, flexible | Performance overhead |
| Network | Graph with records/links | Efficient navigation | Complex structure |

Relational Data Model:

- **Structure:** Data tables (relations) માં organize થાય છે.
- **Components:** Tuples (rows), attributes (columns), domains.
- **Operations:** Select, project, join operations ઉપલબ્ધ છે.

Network Data Model:

- **Structure:** Graph-based જેમાં owner-member relationships હોય છે.
- **Navigation:** Record types વચ્ચે explicit links હોય છે.
- **Flexibility:** Many-to-many relationships ને naturally support કરે છે.

મેમરી ટ્રીક

“HNROE - Have Network Relational Object Entity”

પ્રશ્ન 2(a) [3 ગુણ]

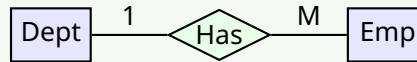
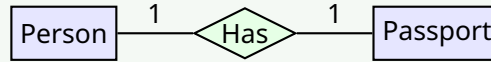
Mapping Cardinalities સમજાવો.

જવાબ

Table:

કોષ્ટક 5. Mapping Cardinalities

| Cardinality | Symbol | Description | Example |
|--------------|--------|---|---------------------|
| One-to-One | 1:1 | દરેક entity એક બીજા entity સાથે જોડાયેલ | Person-Passport |
| One-to-Many | 1:M | એક entity અનેક સાથે જોડાયેલ | Department-Employee |
| Many-to-One | M:1 | અનેક entities એક સાથે જોડાયેલ | Student-Course |
| Many-to-Many | M:N | અનેક entities અનેક સાથે જોડાયેલ | Student-Subject |



આકૃતિ 3. Cardinality Examples

- **Cardinality constraints:** relationship participation limits નક્કી કરે છે.
- **Maximum cardinality:** associations ની upper bound દર્શાવે છે.
- **Importance:** Database design અને relationship modeling માં મદદરૂપ છે.

મેમરી ટ્રીક

“OMOM - One, One-Many, One-Many, Many-Many”

પ્રશ્ન 2(b) [4 ગુણ]

Relational Algebra માં Outer Join operation સમજાવો.

જવાબ

Table:

કોષ્ટક 6. Outer Join Types

| Join Type | Symbol | Result | NULL Handling |
|-------------|-------------|---------------------------|----------------------------|
| Left Outer | \bowtie_L | All left + matching right | Unmatched right માટે NULLs |
| Right Outer | \bowtie_R | All right + matching left | Unmatched left માટે NULLs |
| Full Outer | \bowtie_F | બંને tables માંથી બધું | Unmatched માટે NULLs |

Example: EMPLOYEE LEFT OUTER JOIN DEPARTMENT

- બધા employees નો સમાવેશ થાય છે.
- જે employees નું department ન હોય ત્યાં NULL values મુકાય છે.
- **Preserves tuples:** Specified relation(s) માંથી unmatched tuples જાળવી રાખે છે.
- **NULL values:** Missing attribute values માટે NULL values વાપરે છે.
- **Usage:** Incomplete data relationships ના reporting માટે ઉપયોગી છે.

મેમરી ટ્રીક

“LRF - Left Right Full outer joins”

પ્રશ્ન 2(c) [7 ગુણ]

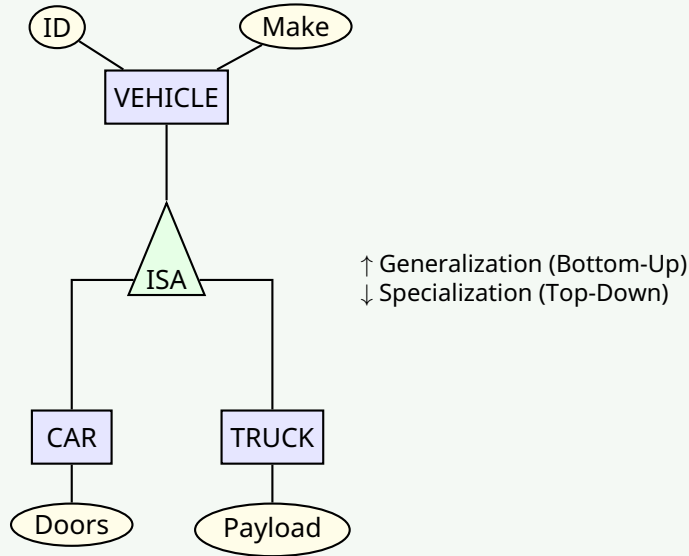
Specialization અને Generalization નો કોન્સેપ્ટ ઉદાહરણ સાથે સમજાવો.

જવાબ

Table:

કોષ્ટક 7. Specialization vs Generalization

| Concept | Direction | Process | Example |
|----------------|-----------|---------------------|----------------------|
| Specialization | Top-Down | General થી Specific | Vehicle → Car, Truck |
| Generalization | Bottom-Up | Specific થી General | Car, Truck → Vehicle |



આકૃતિ 4. ISA Hierarchy Example

Specialization:

- **Process:** Superclass માંથી subclasses બનાવવાની પ્રક્રિયા (Top-Down).
- **Inheritance:** Subclasses બધા superclass attributes inherit કરે છે.
- **Extension:** Subclasses ને specific properties હોય છે.

Generalization:

- **Process:** Common subclass features માંથી superclass બનાવવાની પ્રક્રિયા (Bottom-Up).
- **Abstraction:** Common attributes અને relationships ને identify કરે છે.
- **Simplification:** Hierarchy દ્વારા complexity ઘટાડે છે.

મેમરી ટ્રીક

"SG-TD-BU - Specialization General-To-Detail, Bottom-Up"

પ્રશ્ન 2(a OR) [3 ગુણ]

Relational Algebra માં વિવિધ પ્રકારના Keys સમજાવો.

જવાબ

Table:

કોષ્ટક 8. Types of Keys

| Key Type | Definition | Uniqueness | Example |
|---------------|--|------------|-------------------|
| Super Key | કોઈપણ attribute set જે uniquely identify કરે | Yes | {ID, Name, Phone} |
| Candidate Key | Minimal super key | Yes | {ID}, {Email} |
| Primary Key | પસંદ કરેલ candidate key | Yes | {StudentID} |
| Foreign Key | References primary key | No | {DeptID} |

- **Super Key:** Tuples ને uniquely identify કરે છે, extra attributes હોઈ શકે છે.
- **Candidate Key:** Redundant attributes વગરની minimal super key.
- **Primary Key:** Entity identification માટે પસંદ કરેલ candidate key.
- **Foreign Key:** Tables વચ્ચે referential integrity સ્થાપિત કરે છે.

મેમરી ટ્રીક

"SCPF - Super Candidate Primary Foreign"

પ્રશ્ન 2(b OR) [4 ગુણ]

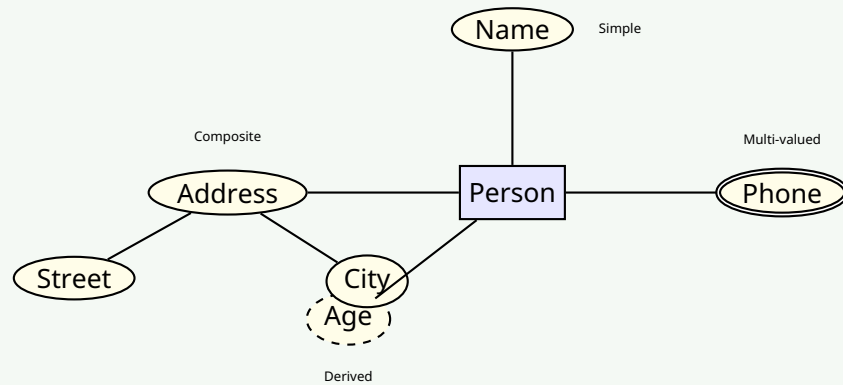
ER-diagram માં attributes ના પ્રકારો યોગ્ય ઉદાહરણ સાથે સમજાવો.

જવાબ

Table:

કોષ્ટક 9. Attribute Types

| Attribute Type | Symbol | Description | Example |
|----------------|-------------|---------------------|------------------------|
| Simple | Oval | વિભાજિત ન થઈ શકે | Age, Name |
| Composite | Tree-like | વિભાજિત થઈ શકે | Address (Street, City) |
| Derived | Dashed oval | અન્ય પરથી ગણતરી થાય | Age from Birth_Date |
| Multi-valued | Double oval | એકથી વધુ values | Phone_Numbers |



આકૃતિ 5. Attribute Types in ER

- **Simple attributes:** atomic અને અવિભાજ્ય હોય છે.
- **Composite attributes:** ને અર્થપૂર્ણ પેટા-ભાગો હોય છે (e.g., Address broken into Street, City).
- **Derived attributes:** અન્ય attribute values પરથી ગણવામાં આવે છે (e.g., Age from DOB).
- **Multi-valued attributes:** entity દીઠ અનેક values store કરી શકે છે (e.g., Phone numbers).

મેમરી ટ્રીક

``SCDM - Simple Composite Derived Multi-valued``

પ્રશ્ન 2(c OR) [7 ગુણ]

SELECT, PROJECT, UNION અને SET-INTERSECTION operations યોગ્ય ઉદાહરણ સાથે સમજાવો.

જવાબ

Table:

કોષ્ટક 10. Relational Operations

| Operation | Symbol | Purpose | Example |
|--------------|----------|-------------------|-------------------------------------|
| SELECT | σ | Filter rows | $\sigma_{salary > 50000}(Employee)$ |
| PROJECT | π | Select columns | $\pi_{name, age}(Employee)$ |
| UNION | \cup | Combine relations | $R \cup S$ |
| INTERSECTION | \cap | Common tuples | $R \cap S$ |

1. SELECT Operation (σ):

- Rows ને filter કરે છે.
- Example: $\sigma_{age > 25}(STUDENT)$
- 25 વર્ષથી વધુ ઉંમરના વિદ્યાર્થીઓ પરત કરે છે.

2. PROJECT Operation (π):

- Specific columns select કરે છે.
- Example: $\pi_{name, course}(STUDENT)$
- ફક્ત name અને course columns પરત કરે છે.

3. UNION Operation (\cup):

- બે relations માંથી tuples ને જોડે છે (duplicates કાઢી નાખે છે).
- Example: $SCIENCE_STUDENTS \cup ARTS_STUDENTS$
- બંને streams ના વિદ્યાર્થીઓને જોડે છે.

4. INTERSECTION Operation (\cap):

- બંને relations માં હાજર હોય તેવા tuples પરત કરે છે.
- Example: $MALE_STUDENTS \cap SPORTS_STUDENTS$
- Male students જે sports રમે છે તેમને પરત કરે છે.

મેમરી ટ્રીક

``SPUI - Select Project Union Intersection``

પ્રશ્ન 3(a) [3 ગુણ]

Primary Key અને Foreign Key constraint વચ્ચેનો તફાવત આપો.

જવાબ

Table:

કોષ્ટક 11. Primary Key vs Foreign Key

| Aspect | Primary Key | Foreign Key |
|-------------|-----------------------|-----------------------|
| Purpose | Unique identification | Referential integrity |
| NULL Values | Not allowed | Allowed |
| Uniqueness | Must be unique | Can be duplicate |
| Count | Only one per table | Multiple allowed |

- **Primary Key:** Table ની અંદર entity integrity સુનિશ્ચિત કરે છે (Unique + Not Null).
- **Foreign Key:** Tables વચ્ચે referential integrity જાળવે છે (links to PK).
- **Uniqueness:** Primary keys હંમેશા unique હોય છે; foreign keys repeat થઈ શકે છે.

મેમરી ટ્રીક

“PU-FN - Primary Unique, Foreign Nullable”

પ્રશ્ન 3(b) [4 ગુણ]

DUAL table અને SYSDATE ને ઉદાહરણ સાથે સમજાવો.

જવાબ

Table:

કોષ્ટક 12. DUAL and SYSDATE

| Component | Type | Purpose | Example |
|-----------|-----------------|-------------------|--------------------------|
| DUAL | Virtual table | Test expressions | SELECT 2+3 FROM DUAL |
| SYSDATE | System function | Current date/time | SELECT SYSDATE FROM DUAL |

DUAL Table:

- **Virtual table:** એક row અને એક column ધરાવતું special table (Oracle).
- **Testing:** System functions અથવા calculations (જે physical table સાથે જોડાયેલ નથી) test કરવા માટે વપરાય છે.

SYSDATE Function:

- **System Time:** Database server પરથી current date અને time પરત કરે છે.
- **Operations:** Date arithmetic support કરે છે (e.g., SYSDATE + 1 આવતીકાલ માટે).

Examples:

- SELECT SYSDATE FROM DUAL; (Current Time)
- SELECT SYSDATE + 30 FROM DUAL; (30 days later)

મેમરી ટ્રીક

“DT-ST - DUAL Testing, SYSDATE Time”

પ્રશ્ન 3(c) [7 ગુણ]

વિવિધ numeric functions નો ઉપયોગ કરવા માટે SQL queries લખો.

જવાબ

Numeric Functions Overview:

- TRUNC: Number ને specified precision સુધી truncate કરે છે.
- ABS: Absolute value શોધી આપે છે.
- CEIL: Smallest integer $\geq n$.

- FLOOR: Largest integer $\leq n$.
- MOD: Remainder (શેષ) શોધી આપે છે.
- POWER: Exponentiation (ઘાત) શોધી આપે છે.

SQL Queries:

- (a) Display integer value of 125.25
SELECT TRUNC(125.25) FROM DUAL; → 125
- (b) Display absolute value of (-15)
SELECT ABS(-15) FROM DUAL; → 15
- (c) Display ceil value of 55.65
SELECT CEIL(55.65) FROM DUAL; → 56
- (d) Display floor value of 100.2
SELECT FLOOR(100.2) FROM DUAL; → 100
- (e) Display the square root of 16
SELECT SQRT(16) FROM DUAL; → 4
- (f) Show value of e^3
SELECT EXP(3) FROM DUAL;
- (g) Display result of 12 raised to 6
SELECT POWER(12, 6) FROM DUAL;
- (h) Display result of 24 mod 2
SELECT MOD(24, 2) FROM DUAL; → 0
- (i) Show output of sign(-25), sign(25), sign(0)
SELECT SIGN(-25), SIGN(25), SIGN(0) FROM DUAL; → -1, 1, 0

મેમરી ટ્રીક

“TACFSEPM - TRUNC ABS CEIL FLOOR SQRT EXP POWER MOD”

પ્રશ્ન 3(a OR) [3 ગુણ]

Unique અને Check Constraint ને યોગ્ય ઉદાહરણ સાથે સમજાવો.

જવાબ

Table:

કોષ્ટક 13. Relationship Constraints

| Constraint | Purpose | Duplicates | Example |
|------------|----------------------|--------------------|---------------|
| UNIQUE | Duplicates અટકાવે છે | Not allowed | Email address |
| CHECK | Data validate કરે છે | Value restrictions | Age > 0 |

Examples:

- **UNIQUE Constraint:**
email VARCHAR(50) UNIQUE ખાતરી કરે છે કે કોઈ બે students ને same email નથી.
- **CHECK Constraint:**
age NUMBER CHECK (age >= 18) ખાતરી કરે છે કે ઉંમર valid છે.

મેમરી ટ્રીક

“UC-DV - Unique no Copy, Check Validates”

પ્રશ્ન 3(b OR) [4 ગુણ]

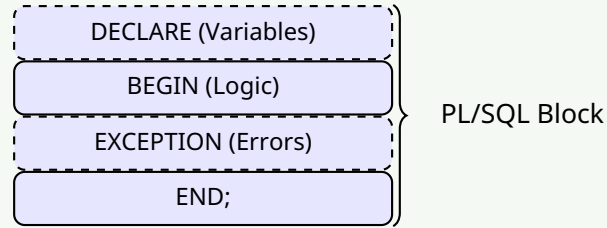
PL/SQL block નું structure સમજાવો.

જવાબ

Table:

કોષ્ટક 14. PL/SQL Block Sections

| Section | Required | Purpose | Example |
|------------------|-----------|-----------------------|----------------------|
| DECLARE | Optional | Variable declarations | v_name VARCHAR2(20); |
| BEGIN | Mandatory | Executable statements | SELECT ... INTO ... |
| EXCEPTION | Optional | Error handling | WHEN OTHERS THEN ... |
| END | Mandatory | Block termination | END; |



આકૃતિ 6. PL/SQL Block Structure

- **DECLARE section:** Variable અને cursor declarations (Optional).
- **BEGIN-END:** Mandatory executable section જેમાં SQL અને logic હોય છે.
- **EXCEPTION section:** Runtime errors ને handle કરે છે (Optional).

મેમરી ટ્રીક

``DBE-E - Declare Begin Exception End``

પ્રશ્ન 3(c OR) [7 ગુણ]

નીચેના ટેબલને ધ્યાનમાં લો અને ક્વેરીઓ ઉકેલો.

જવાબ

I) BRANCH table બનાવો:

```

1 CREATE TABLE BRANCH (
2   branchid VARCHAR2(10) PRIMARY KEY,
3   branchname VARCHAR2(50) NOT NULL,
4   address VARCHAR2(100)
5 );
  
```

II) EMPLOYEE table બનાવો:

```

1 CREATE TABLE EMPLOYEE (
2   empid VARCHAR2(10) PRIMARY KEY,
3   name VARCHAR2(50) NOT NULL,
4   post VARCHAR2(30),
5   gender CHAR(1) CHECK (gender IN ('M', 'F')),
6   birthdate DATE,
7   salary NUMBER(10,2),
8   branchid VARCHAR2(10),
9   FOREIGN KEY (branchid) REFERENCES BRANCH(branchid)
10 );
  
```

III) અમદાવાદ બ્રાન્ચમાં કામ કરતા Employees શોધો:

```

1 SELECT e.* FROM EMPLOYEE e, BRANCH b
2 WHERE e.branchid = b.branchid
3 AND b.branchname = 'Ahmedabad';

```

IV) 1998 માં જન્મેલા Employees શોધો:

```

1 SELECT * FROM EMPLOYEE
2 WHERE EXTRACT(YEAR FROM birthdate) = 1998;

```

V) 5000 થી વધુ પગાર ધરાવતી Female employees શોધો:

```

1 SELECT * FROM EMPLOYEE
2 WHERE gender = 'F' AND salary > 5000;

```

VI) જ્યાં Ajay કામ કરે છે તે Address શોધો:

```

1 SELECT b.address FROM EMPLOYEE e, BRANCH b
2 WHERE e.branchid = b.branchid
3 AND e.name = 'Ajay';

```

મેમરી ટ્રીક

``CBEFFA - Create Branch Employee Find Female Address"

પ્રશ્ન 4(a) [3 ગુણ]

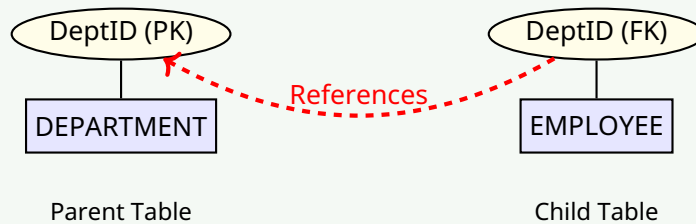
Referential Integrity યથા યોગ્ય ઉદાહરણ સાથે સમજાવો.

જવાબ

Table:

કોષ્ટક 15. Referential Integrity

| Aspect | Description | Example |
|------------|---|-------------------------------------|
| Definition | Foreign key must reference existing primary key | Employee.deptid → Department.deptid |
| Purpose | Maintain data consistency | Prevent orphan records |
| Actions | CASCADE, SET NULL, RESTRICT | ON DELETE CASCADE |



આકૃતિ 7. Foreign Key Reference

- **Referential integrity:** tables વચ્ચે valid relationships સુનિશ્ચિત કરે છે.
- **Orphan records:** prevented; અસ્તિત્વમાં ન હોય તેવા department માં employee ઉમેરી શકાતો નથી.
- **Cascade:** operations updates/deletes ને automatically propagate કરે છે.

મેમરી ટ્રીક

“RIO - Referential Integrity prevents Orphans”

પ્રશ્ન 4(b) [4 ગુણ]

Partial અને Full Functional Dependency વચ્ચેનો તફાવત આપો.

જવાબ

Table:

કોષ્ટક 16. Dependency Types

| Type | Definition | Example | Requirement |
|---------|--------------------------------|-------------------------------|-----------------------|
| Partial | Composite key ના ભાગ પર આધારિત | (StudentID, CourseID) → Name | Composite primary key |
| Full | આખી key પર આધારિત | (StudentID, CourseID) → Grade | Complete key needed |

Example: Key is (StudentID, CourseID)

- **Partial:** StudentName only StudentID પર આધારિત છે. 2NF નું ઉલ્લંઘન કરે છે.
- **Full:** Grade બંને StudentID અને CourseID પર આધારિત છે. Good.
- **Partial dependency** data redundancy અને anomalies નું કારણ બને છે.
- 2NF partial functional dependencies ને દૂર કરે છે.

મેમરી ટ્રીક

“PF-CF - Partial Few, Complete Full”

પ્રશ્ન 4(c) [7 ગુણ]

3rd Normal Form ઉદાહરણ સાથે સમજાવો.

જવાબ

3rd Normal Form Requirements:

- Table 2NF માં હોવું જોઈએ.
- કોઈ transitive dependencies ન હોવી જોઈએ (Non-key → Non-key).
- Non-key attributes માત્ર primary key પર આધારિત હોવા જોઈએ.

Problem (Before 3NF):

| StudID | Name | InstID | InstName |
|--------|------|--------|-----------|
| S1 | John | I1 | Dr. Smith |

Issue: StudentID → InstID → InstName (Transitive Dependency)

3NF Solution (Decomposition):

1. STUDENT Table:

| StudID | Name | CourseID |
|--------|------|----------|
| S1 | John | C1 |
2. COURSE Table:

| CourseID | CName | InstID |
|----------|-------|--------|
| C1 | Math | I1 |
3. INSTRUCTOR Table:

| InstID | InstName |
|--------|-----------|
| I1 | Dr. Smith |

મેમરી ટ્રીક

``3NF-NT - 3rd Normal Form No Transitives"

પ્રશ્ન 4(a OR) [3 ગુણ]

Normalization નું મહત્વ સમજાવો.

જવાબ

Table:

કોષ્ટક 17. Benefits of Normalization

| Benefit | Problem Solved | Result |
|---------------------|-----------------------------|----------------------|
| Reduce Redundancy | Duplicate data | Storage efficiency |
| Eliminate Anomalies | Update/Insert/Delete issues | Data consistency |
| Improve Integrity | Data inconsistency | Reliable information |

- **Minimizes redundancy:** Data એક જ વાર store કરીને storage બચાવે છે.
- **Prevents anomalies:** Logical data modification સુનિશ્ચિત કરે છે.
- **Simplifies maintenance:** Organized structure manage કરવું સરળ છે.

મેમરી ટ્રીક

``RESIM - Redundancy Eliminated, Storage Improved, Maintenance"

પ્રશ્ન 4(b OR) [4 ગુણ]

Prime Attributes અને Non-Prime Attributes વચ્ચેનો તફાવત આપો.

જવાબ

Table:

કોષ્ટક 18. Prime vs Non-Prime Attributes

| Type | Definition | Role | Example |
|-----------|-----------------------|---------------|---------------------|
| Prime | Part of candidate key | Key formation | StudentID, CourseID |
| Non-Prime | Not part of any key | Data storage | Name, Grade |

Example: Relation(StudentID, CourseID, Grade, Semester)

- **Candidate Key:** (StudentID, CourseID)
- **Prime Attributes:** StudentID, CourseID
- **Non-Prime Attributes:** Grade, Semester
- **Significance:** Non-prime attributes સાથે સંકળાયેલી functional dependency Normal Form (2NF/3NF) નક્કી કરે છે.

મેમરી ટ્રીક

``PN-KD - Prime in Key, Non-prime for Data"

પ્રશ્ન 4(c OR) [7 ગુણ]

2nd Normal Form ઉદાહરણ સાથે સમજાવો.

જવાબ

2nd Normal Form Requirements:

- Table 1NF માં હોવું જોઈએ.
- કોઈ partial functional dependencies ન હોવી જોઈએ.
- Non-key attributes એ WHOLE primary key પર આધારિત હોવા જોઈએ.

Problem (Before 2NF):

| StudID | CourseID | SName | CName | Grade |
|--------|----------|-------|---------|-------|
| S1 | C1 | John | Math | A |
| S1 | C2 | John | Physics | B |

Issue: StudID → SName (Partial), CourseID → CName (Partial)

2NF Solution (Decomposition):

- STUDENT Table:**

| StudID | SName |
|--------|-------|
| S1 | John |
- COURSE Table:**

| CourseID | CName |
|----------|-------|
| C1 | Math |
- ENROLLMENT Table:**

| StudID | CourseID | Grade |
|--------|----------|-------|
| S1 | C1 | A |
| S1 | C2 | B |

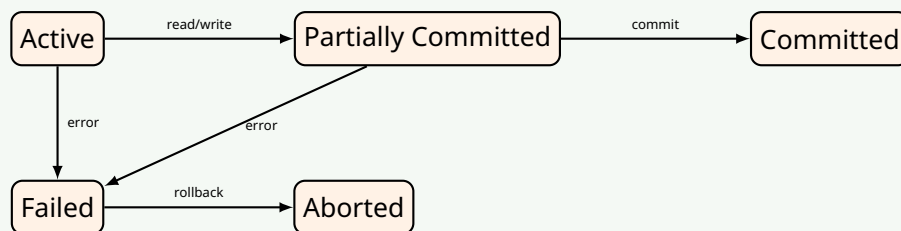
મેમરી ટ્રીક

"2NF-FD - 2nd Normal Form Full Dependencies"

પ્રશ્ન 5(a) [3 ગુણ]

Transaction states યોગ્ય ડાયાગ્રામ સાથે સમજાવો.

જવાબ



આકૃતિ 8. Transaction State Diagram

Table:

કોષ્ટક 19. Transaction States

| State | Description | Next State |
|---------------------|----------------------------------|------------------|
| Active | Operations execute કરી રહ્યું છે | Partial/Failed |
| Partially Committed | છેલ્લું statement execute થયું | Committed/Failed |
| Committed | સફળતાપૂર્વક પૂર્ણ | End |
| Failed | Error ને કારણે execution અટક્યું | Aborted |
| Aborted | Changes rolled back | End |

મેમરી ટ્રીક

“APCFA - Active Partial Commit Fail Abort”

પ્રશ્ન 5(b) [4 ગુણ]

કોઈપણ બે DDL commands ને યોગ્ય ઉદાહરણ સાથે સમજાવો.

જવાબ

Table:

કોષ્ટક 20. DDL Commands

| Command | Purpose | Example Syntax |
|---------|------------------|-----------------------|
| CREATE | Create objects | CREATE TABLE T (...) |
| ALTER | Modify structure | ALTER TABLE T ADD ... |

1. CREATE Command:

```
1 CREATE TABLE EMPLOYEE (
2   empid NUMBER(5) PRIMARY KEY,
3   name VARCHAR2(50)
4 );
```

2. ALTER Command:

```
1 -- Add new column
2 ALTER TABLE EMPLOYEE ADD phone VARCHAR2(15);
3 -- Modify column
4 ALTER TABLE EMPLOYEE MODIFY name VARCHAR2(100);
5 -- Drop column
6 ALTER TABLE EMPLOYEE DROP COLUMN phone;
```

- **DDL:** Data Definition Language schema વ્યાખ્યાયિત કરે છે.
- **Auto-commit:** DDL commands આપમેળે save થાય છે.

મેમરી ટ્રીક

“CA-NM - CREATE Adds, ALTER Modifies”

પ્રશ્ન 5(c) [7 ગુણ]

ACID ગુણધર્મો વિગતવાર સમજાવો.

જવાબ

Table:

કોષ્ટક 21. ACID Properties

| Property | Definition | Example |
|-------------|-----------------------|----------------------------------|
| Atomicity | બધું અથવા કંઈ નહીં | Complete transfer અથવા rollback |
| Consistency | Database હંમેશા valid | Balance ક્યારેય negative ન થાય |
| Isolation | Independent execution | Concurrent users interfere ન કરે |
| Durability | Permanent changes | Power loss પછી પણ data remains |

- **Atomicity:** Transaction એક indivisible unit છે.
- **Consistency:** DB ને એક valid state માંથી બીજામાં લઈ જાય છે.
- **Isolation:** Intermediate states અન્ય transactions માટે invisible હોય છે.
- **Durability:** Committed data crash પછી પણ કાયમી રહે છે.

મેમરી ટ્રીક

"ACID - Atomicity Consistency Isolation Durability"

પ્રશ્ન 5(a OR) [3 ગુણ]

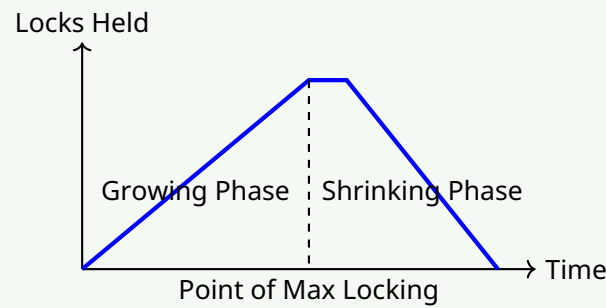
two phase લોકિંગ ટેકનિક શું છે?

જવાબ

Table:

કોષ્ટક 22. 2PL Phases

| Phase | Action | Description | Allowed |
|-----------|---------------|--------------------------------|---------|
| Growing | Acquire locks | Transaction બધા locks મેળવે છે | LOCK |
| Shrinking | Release locks | Transaction locks છોડે છે | UNLOCK |



આકૃતિ 9. Two-Phase Locking Protocol

- **Growing Phase:** Transaction locks મેળવી શકે છે, પણ કોઈ release કરી શકતું નથી.
- **Shrinking Phase:** Transaction locks release કરી શકે છે, પણ કોઈ મેળવી શકતું નથી.
- **Guarantee:** Serializability સુનિશ્ચિત કરે છે.

મેમરી ટ્રીક

"2PL-GS - Two Phase Locking Growing Shrinking"

પ્રશ્ન 5(b OR) [4 ગુણ]

કોઈપણ બે DML આદેશોને યોગ્ય ઉદાહરણ સાથે સમજાવો.

જવાબ

Table:

કોષ્ટક 23. DML Commands

| Command | Purpose | Example |
|---------|-----------------|------------------------|
| INSERT | Add new records | INSERT INTO Student... |
| UPDATE | Modify records | UPDATE Student SET... |

1. INSERT Command:

```
1 INSERT INTO EMPLOYEE (empid, name, salary)
2 VALUES (101, 'John Smith', 50000);
```

2. UPDATE Command:

```
1 UPDATE EMPLOYEE
2 SET salary = 60000
3 WHERE empid = 101;
```

- **DML:** Data Manipulation Language data values ને manage કરે છે.
- **Explicit Commit:** Changes save કરવા માટે ઘણીવાર COMMIT ની જરૂર પડે છે.

મેમરી ટ્રીક

“IU-AM - INSERT Adds, UPDATE Modifies”

પ્રશ્ન 5(c OR) [7 ગુણ]

concurrency control ની સમસ્યાઓની યાદી બનાવો અને કોઈપણ બેને વિગતવાર સમજાવો.

જવાબ

Problems List:

1. Lost Update Problem
2. Dirty Read Problem
3. Non-repeatable Read Problem
4. Phantom Read Problem

1. Lost Update Problem:

- **Scenario:** બે transactions same item update કરે છે, અને એક update બીજા દ્વારા overwrite થાય છે.
- **Example:**
 - T1 reads X=100. T2 reads X=100.
 - T1 writes X=110. T2 writes X=90.
 - T1 નું update ખોવાઈ ગયું. Final X=90 (હોવું જોઈએ 100).

2. Dirty Read Problem:

- **Scenario:** Uncommitted transaction દ્વારા લખાયેલ data વાંચવો.
- **Example:**
 - T1 updates X=200. T2 reads X=200.
 - T1 fails and rolls back X=100.
 - T2 એ invalid data (200) process કર્યો.

Solutions:

- Use **Locking Protocols** (2PL).
- Use **Isolation Levels** (Read Committed, Serializable).

મેમરી ટ્રીક

“LDUI - Lost Dirty Unrepeatable Inconsistent”