

# ડેટાબેઝ મેનેજમેન્ટ (4331603) - શિયાળો 2023 સોલ્યુશન

Milav Dabgar

January 18, 2024

## પ્રશ્ન 1(a) [3 ગુણ]

નીચેના શબ્દો વ્યાખ્યાયિત કરો: a). Data items b). Data dictionary c). Meta data

જવાબ

ટેબલ:

કોષ્ટક 1. ડેટાબેઝ શબ્દો

શબ્દ	વ્યાખ્યા
Data Items	ડેટાના મૂળભૂત એકમો જે વધુ વિભાજન કરી શકાતા નથી. ડેટાબેઝ ફીલ્ડ્સમાં સંગ્રહિત વ્યક્તિગત તથ્યો અથવા મૂલ્યો
Data Dictionary	ડેટાબેઝ સ્ટ્રક્ચર, ટેબલ્સ, કોલમ્સ અને સંબંધો વિશે મેટાડેટા ધરાવતો કેન્દ્રીય ભંડાર
Metadata	ડેટા વિશેનો ડેટા જે ડેટાબેઝ એલિમેન્ટ્સની રચના, અવરોધો અને ગુણધર્મોનું વર્ણન કરે છે

- **Data Items:** ડેટાનો સૌથી નાનો એકમ
- **Data Dictionary:** સ્કીમા વ્યાખ્યાઓ સંગ્રહિત કરતું "સિસ્ટમ કેટલોગ"
- **Metadata:** ડેટાનું વર્ણન કરતી માળખાકીય માહિતી

મેમરી ટ્રીક

"DDM - Data Dictionary Manages"

## પ્રશ્ન 1(b) [4 ગુણ]

ફાઇલ ઓરિએન્ટેડ સિસ્ટમના ગેરફાયદા સમજાવો.

જવાબ

ટેબલ:

કોષ્ટક 2. ફાઇલ સિસ્ટમના ગેરફાયદા

ગેરફાયદો	વિવરણ
ડેટા રીડ-ડ-સી	બહુવિધ ફાઇલોમાં સમાન ડેટાનો સંગ્રહ, જે સ્ટોરેજનો બગાડ કરે છે
ડેટા અસંગતતા	વિવિધ ફાઇલોમાં સમાન ડેટાના વિવિધ વર્ઝન
ડેટા આઇસોલેશન	બહુવિધ ફાઇલોમાં વિખરાયેલા ડેટાને એક્સેસ કરવામાં મુશ્કેલી
સિક્યોરિટી સમસ્યાઓ	મર્યાદિત એક્સેસ કંટ્રોલ અને સુરક્ષા મિકેનિઝમ

- **રીડ-ડ-સી:** ફાઇલોમાં ડેટાનું ડુપ્લિકેશન
- **અસંગતતા:** નબળા સિંક્રનાઇઝેશનને કારણે મિસમેચ ડેટા
- **આઇસોલેશન:** પ્રમાણભૂત ફોર્મેટનો અભાવ ડેટા શેરિંગને અવરોધે છે

- ઇન્ટેગ્રિટી: ફાઇલોમાં કન્સ્ટ્રેઇન્ટ્સ લાગુ કરવું મુશ્કેલ

### મેમરી ટ્રીક

“RDIS - Really Difficult Information System”

## પ્રશ્ન 1(c) [7 ગુણ]

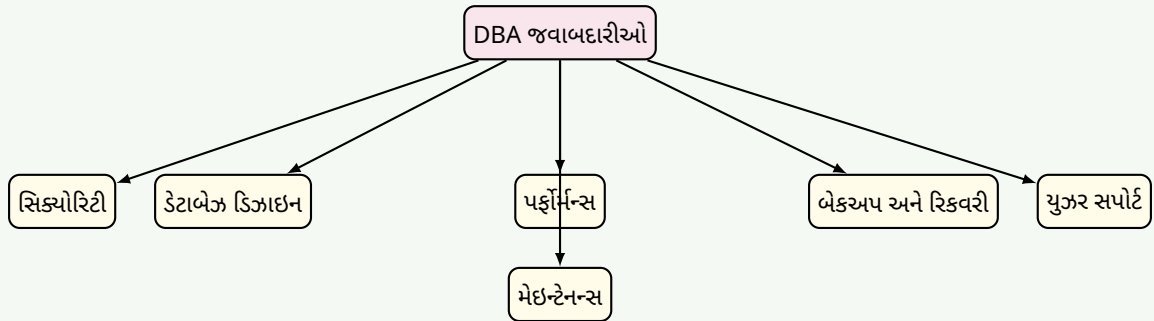
DBA ની જવાબદારીઓનું વિગતવાર વર્ણન કરો.

### જવાબ

ટેબલ:

કોષ્ટક 3. DBA ની જવાબદારીઓ

જવાબદારી	વિગતો
ડેટાબેઝ ડિઝાઇન	લોજિકલ અને ફિઝિકલ ડેટાબેઝ સ્ટ્રક્ચર બનાવવું
સિક્યોરિટી મેનેજમેન્ટ	યુઝર એક્સેસ કંટ્રોલ અને ડેટા પ્રોટેક્શન લાગુ કરવું
પર્ફોર્મન્સ મોનિટરિંગ	ડેટાબેઝ પર્ફોર્મન્સ અને ક્વેરી એક્ઝિક્યુશન ઓપ્ટિમાઇઝ કરવું
બેકઅપ અને રિકવરી	નિયમિત બેકઅપ દ્વારા ડેટા સેફ્ટી સુનિશ્ચિત કરવી
યુઝર સપોર્ટ	ડેટાબેઝ યુઝર્સને ટેકનિકલ સહાય પૂરી પાડવી
સિસ્ટમ મેઇન્ટેનન્સ	નિયમિત અપડેટ્સ, પેચેસ અને સિસ્ટમ ઓપ્ટિમાઇઝેશન



આકૃતિ 1. DBA ની મુખ્ય જવાબદારીઓ

- ડિઝાઇન: સ્કીમા વ્યાખ્યા અને સ્ટોરેજ પ્લાનિંગ
- સિક્યોરિટી: એક્સેસ આપવી/લેવી અને એન્ક્રિપ્શન
- રિકવરી: ડિઝાસ્ટર રિકવરી પ્લાનિંગ
- ટ્યુનિંગ: ઇન્ડેક્સિંગ અને ક્વેરી ઓપ્ટિમાઇઝેશન

### મેમરી ટ્રીક

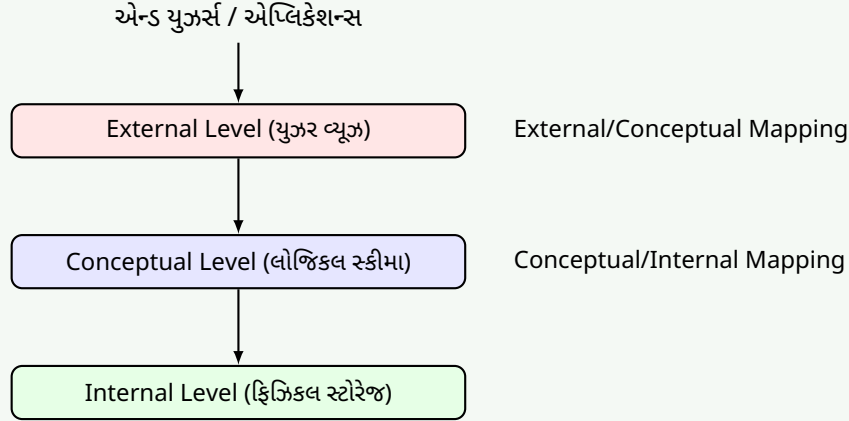
“DSPBUM - Database Specialists Provide Better User Management”

## પ્રશ્ન 1(c OR) [7 ગુણ]

Data abstraction ની વ્યાખ્યા આપો? DBMS નું ત્રિ સ્તરનું આર્કિટેક્ચર સમજાવો.

## જવાબ

**Data Abstraction:** યુઝર્સને માત્ર આવશ્યક ફીચર્સ દર્શાવતી વખતે જટિલ implementation વિગતો છુપાવવાની પ્રક્રિયા.



આકૃતિ 2. ત્રિ સ્તરનું આર્કિટેક્ચર

ટેબલ:

કોષ્ટક 4. આર્કિટેક્ચર સ્તરો

સ્તર	વિવરણ	હેતુ
External Level	ડેટાબેઝનો યુઝર વ્યૂ	વ્યક્તિગત યુઝર પરસ્પેક્ટિવ્સ
Conceptual Level	સંપૂર્ણ ડેટાબેઝની લોજિકલ સ્ટ્રક્ચર	એકંદર ડેટાબેઝ ઓર્ગેનાઇઝેશન
Internal Level	ફિઝિકલ સ્ટોરેજ વિગતો	ડેટા ખરેખર કેવી રીતે સ્ટોર થાય છે

- **View Level:** ડેટાબેઝના માત્ર એક ભાગનું વર્ણન કરે છે
- **Logical Level:** કયો ડેટા સંગ્રહિત છે અને સંબંધોનું વર્ણન કરે છે
- **Physical Level:** જટિલ નિમ્ન-સ્તરની ડેટા સ્ટ્રક્ચર્સનું વર્ણન કરે છે

## મેમરી ટ્રીક

"ECI - Every Computer Industry"

## પ્રશ્ન 2(a) [3 ગુણ]

નીચેના શબ્દો વ્યાખ્યાયિત કરો: a).Relationship set b).Participation c).Candidate key

## જવાબ

ટેબલ:

કોષ્ટક 5. શબ્દાવલી

શબ્દ	વ્યાખ્યા
Relationship Set	એન્ટિટી સેટ્સ વચ્ચે સમાન પ્રકારના સંબંધોનો સંગ્રહ
Participation	અવરોધ જે સ્પષ્ટ કરે છે કે એન્ટિટી ઓકરન્સ સંબંધમાં ફરજિયાત છે કે નહીં
Candidate Key	એટ્રિબ્યુટ્સનો ન્યૂનતમ સેટ જે એન્ટિટી સેટમાં દરેક એન્ટિટીને અનન્ય રીતે ઓળખે છે

- **Relationship:** અનેક એન્ટિટીઝ વચ્ચેનું જોડાણ
- **Total Participation:** દરેક એન્ટિટીએ ભાગ લેવો જ જોઈએ (ડબલ લાઇન)
- **Candidate Key:** સુપર કી જેનો કોઈ પ્રોપર સબસેટ સુપર કી નથી

## મેમરી ટ્રીક

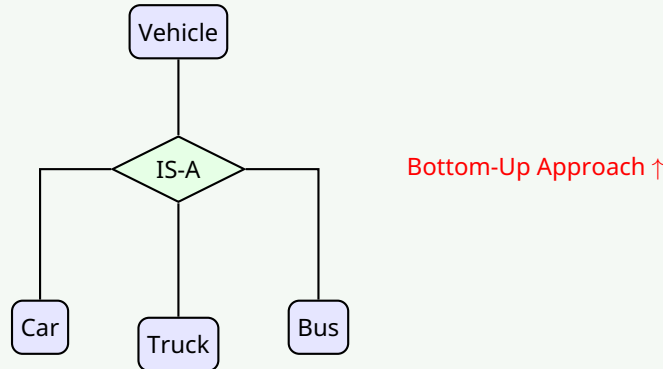
“RPC - Relationship Participation Candidate”

## પ્રશ્ન 2(b) [4 ગુણ]

Generalization ઉદાહરણ સાથે સમજાવો.

## જવાબ

**Generalization:** બોટમ-અપ અપ્રોચ જ્યાં નીચલા-સ્તરની એન્ટિટીઝના સામાન્ય એટ્રિબ્યુટ્સને ઉચ્ચ-સ્તરની એન્ટિટીમાં જોડવામાં આવે છે.



આકૃતિ 3. Generalization ઉદાહરણ

ટેબલ:

કોષ્ટક 6. Generalization ખ્યાલો

ખ્યાલ	વિવરણ
હેતુ	સામાન્ય સુપરકલાસ બનાવીને રીડન્ડન્સી ઘટાડવી
દિશા	બોટમ-અપ (વિશિષ્ટથી સામાન્ય)
ઉદાહરણ	Car, Truck, Bus → Vehicle

- **Superclass:** સામાન્ય ગુણધર્મો ધરાવતી સામાન્યીકૃત એન્ટિટી
- **Subclass:** અનન્ય ગુણધર્મો ધરાવતી વિશિષ્ટ એન્ટિટીઝ
- **Attribute Inheritance:** સબકલાસીસ સુપરકલાસના ગુણધર્મો વારસામાં મેળવે છે

## મેમરી ટ્રીક

“GBU - Generalization Builds Up”

## પ્રશ્ન 2(c) [7 ગુણ]

E-R Diagram ની વ્યાખ્યા આપો? E-R ડાયાગ્રામમાં વપરાતા વિવિધ Symbols ને ઉદાહરણ સાથે સમજાવો.

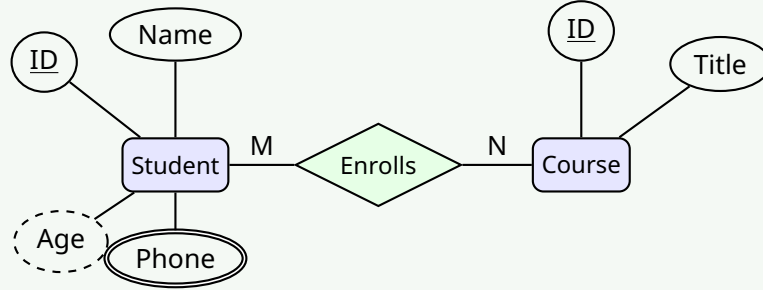
## જવાબ

**E-R Diagram:** ડેટાબેઝ ડિઝાઇનમાં એન્ટિટીઝ, એટ્રિબ્યુટ્સ અને સંબંધો દર્શાવતું ગ્રાફિકલ પ્રતિનિધિત્વ.

ટેબલ:

કોષ્ટક 7. ER ડાયાગ્રામ સિમ્બોલ્સ

સિમ્બોલ	આકાર	ઉપયોગ	ઉદાહરણ
Entity	લંબચોરસ	ઓબ્જેક્ટ્સનું પ્રતિનિધિત્વ	Student, Course
Attribute	અંડાકાર	એન્ટિટીઝના ગુણધર્મો	Name, Age, ID
Relationship	ડાયમંડ	એન્ટિટીઝ વચ્ચેના જોડાણો	Enrolls, Teaches
Primary Key	અન્ડરલાઇન્ડ અંડાકાર	અનન્ય ઓળખકર્તા	Student_ID
Multivalued	ડબલ અંડાકાર	બહુવિધ મૂલ્યો	Phone_Numbers
Derived	ડેશ અંડાકાર	ગણતરી કરેલા એટ્રિબ્યુટ્સ	Age from DOB



આકૃતિ 4. ER ડાયાગ્રામ ઉદાહરણ

- **Entity Sets:** સ્વતંત્ર અસ્તિત્વ ધરાવતી વસ્તુઓ
- **Relationship Sets:** એન્ટિટીઝ વચ્ચેના જોડાણો
- **Attribute Types:** Key, Composite, Multivalued, Derived

## મેમરી ટ્રીક

“EARPM - Every Attribute Represents Proper Meaning”

## પ્રશ્ન 2(a OR) [3 ગુણ]

Relational Algebra ની વ્યાખ્યા આપો? Relational Algebra માં વિવિધ કામગીરીની યાદી આપો?

## જવાબ

**Relational Algebra:** રિલેશનલ ડેટાબેઝ ટેબલ્સને મેનિપ્યુલેટ કરવા માટેની ઓપરેશન્સ સાથે ફોર્મલ ક્વેરી લેંગ્વેજ.  
ટેબલ:

કોષ્ટક 8. Relational Algebra ઓપરેશન્સ

ઓપરેશન પ્રકાર	ઓપરેશન્સ
મૂળભૂત ઓપરેશન્સ	Select ( $\sigma$ ), Project ( $\pi$ ), Union ( $\cup$ ), Set Difference ( $-$ ), Cartesian Product ( $\times$ )
વધારાની ઓપરેશન્સ	Intersection ( $\cap$ ), Join ( $\bowtie$ ), Division ( $\div$ ), Rename ( $\rho$ )

- **Procedural Language:** શું મેળવવું અને કેવી રીતે મેળવવું તે સ્પષ્ટ કરે છે
- **Operators:** રિલેશન્સ ઇનપુટ તરીકે લે છે અને આઉટપુટ તરીકે રિલેશન આપે છે

## મેમરી ટ્રીક

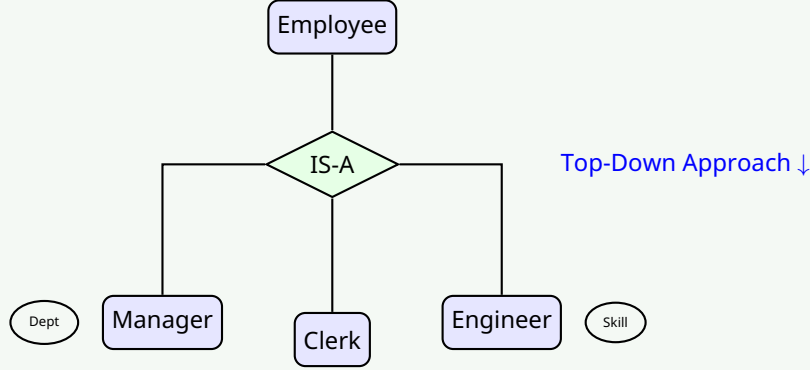
“SPUDC-IJDR - Simple People Use Database Concepts”

## પ્રશ્ન 2(b OR) [4 ગુણ]

Specialization ઉદાહરણ સાથે સમજાવો.

જવાબ

**Specialization:** ટોપ-ડાઉન અપ્રોચ જ્યાં ઉચ્ચ-સ્તરની એન્ટિટીને વિશિષ્ટ નીચલા-સ્તરની એન્ટિટીઝમાં વિભાજિત કરવામાં આવે છે.



આકૃતિ 5. Specialization ઉદાહરણ

ટેબલ:

કોષ્ટક 9. Specialization ખ્યાલો

ખ્યાલ	વિવરણ
હેતુ	અનન્ય એટ્રિબ્યુટ્સ સાથે વિશિષ્ટ સબકલાસીસ બનાવવી
દિશા	ટોપ-ડાઉન (સામાન્યથી વિશિષ્ટ)
ઉદાહરણ	Employee → Manager, Clerk, Engineer

- **Subgrouping:** વિશિષ્ટ ભૂમિકાઓ સાથે એન્ટિટીઝના સબસેટ્સ ઓળખે છે
- **Inheritance:** નીચલા-સ્તરની એન્ટિટીઝ ઉચ્ચ-સ્તરથી વારસો મેળવે છે

મેમરી ટ્રીક

“STD - Specialization Top Down”

## પ્રશ્ન 2(c OR) [7 ગુણ]

Attribute ની વ્યાખ્યા આપો? વિવિધ પ્રકારના Attribute ને ઉદાહરણ સાથે સમજાવો.

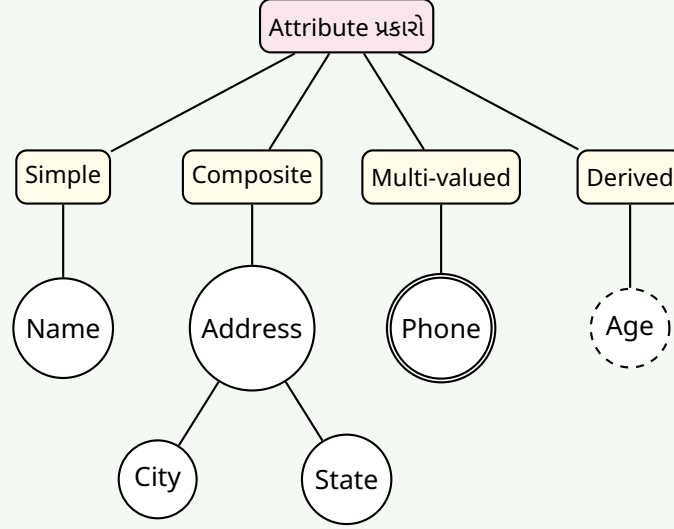
જવાબ

**Attribute:** એન્ટિટીનું વર્ણન કરતી મિલકત અથવા લાક્ષણિકતા.

ટેબલ:

કોષ્ટક 10. Attribute પ્રકારો

એટ્રિબ્યુટ પ્રકાર	વિવરણ	ઉદાહરણ
Simple	વધુ વિભાજન કરી શકાતું નથી	Age, Name
Composite	ઉપવિભાગ કરી શકાય છે	Address (Street, City)
Single-valued	એક મૂલ્ય ધરાવે છે	SSN, Employee_ID
Multi-valued	બહુવિધ મૂલ્યો હોઈ શકે છે	Phone_Numbers
Derived	અન્ય એટ્રિબ્યુટ્સથી ગણતરી કરેલ	Age from Birth_Date
Key	એન્ટિટીને અનન્ય રીતે ઓળખે છે	Student_ID



આકૃતિ 6. એટ્રિબ્યુટ્સનું વર્ગીકરણ

- **Domain:** દરેક ડેટા માટે માન્ય મૂલ્યોનો સમૂહ
- **Null:** મૂલ્ય જ્યારે અજ્ઞાત હોય અથવા લાગુ પડતું નથી

## મેમરી ટ્રીક

“SCSMDK - Simple Composite Single Multi Derived Key”

## પ્રશ્ન 3(a) [3 ગુણ]

SQL માં GRANT અને REVOKE સ્ટેટમેન્ટ સમજાવો.

## જવાબ

ટેબલ:

કોષ્ટક 11. DCL કમાન્ડ્સ

સ્ટેટમેન્ટ	હેતુ	સિન્ટેક્સ ઉદાહરણ
GRANT	યુઝર્સને વિશેષાધિકારો પ્રદાન કરે છે	GRANT SELECT ON table TO user
REVOKE	યુઝર્સ પાસેથી વિશેષાધિકારો દૂર કરે છે	REVOKE SELECT ON table FROM user

- **Privileges:** પરવાનગીઓ જેમ કે SELECT, INSERT, UPDATE, DELETE
- **Control:** કોણ ડેટા એક્સેસ કરી શકે છે તેનું સંચાલન
- **Security:** ડેટાબેઝ સુરક્ષા માટે મૂળભૂત

## મેમરી ટ્રીક

``GR - Grant Removes (via REVOKE)``

## પ્રશ્ન 3(b) [4 ગુણ]

નીચેના Character function સમજાવો .1) INSTR 2) LENGTH

## જવાબ

ટેબલ:

કોષ્ટક 12. SQL Character ફંક્શન્સ

ફંક્શન	હેતુ	સિન્ટેક્સ	ઉદાહરણ
INSTR	સબસ્ટ્રિંગની સ્થિતિ શોધે છે	INSTR(str, substr)	INSTR('Hello', 'e') → 2
LENGTH	સ્ટ્રિંગની લંબાઈ રિટર્ન કરે છે	LENGTH(str)	LENGTH('Hello') → 5

```
1 SELECT INSTR('Database', 'a') FROM dual; -- Returns 2
2 SELECT LENGTH('Database') FROM dual; -- Returns 8
```

- **INSTR**: સબસ્ટ્રિંગ માટે કેસ સેન્સિટિવ શોધ
- **LENGTH**: સ્પેસ સહિત કુલ અક્ષરોની ગણતરી કરે છે

## મેમરી ટ્રીક

``IL - INSTR Locates, LENGTH measures``

## પ્રશ્ન 3(c) [7 ગુણ]

નીચેના Table માટે SQL સ્ટેટમેન્ટ લખો: Student(Enno,name,branch,sem,clgname,bdate)

## જવાબ

```
1 -- 1. Create a table Student
2 CREATE TABLE Student (
3   Enno VARCHAR(10) PRIMARY KEY,
4   name VARCHAR(50),
5   branch VARCHAR(20),
6   sem INT,
7   clgname VARCHAR(100),
8   bdate DATE
9 );
10
11 -- 2. Add a column mobno in Student table
12 ALTER TABLE Student ADD mobno VARCHAR(15);
13
14 -- 3. Insert one record in student table
15 INSERT INTO Student VALUES
16 ('E001', 'Raj Patel', 'IT', 3, 'GTU College', '2003-05-15', '9876543210');
17
18 -- 4. Find out list of students who have enrolled in "IT" branch
19 SELECT * FROM Student WHERE branch = 'IT';
20
```



```

21 -- 5. Retrieve all information about student where name begin with 'a'
22 SELECT * FROM Student WHERE name LIKE 'a%';
23
24 -- 6. Count the number of rows in student table
25 SELECT COUNT(*) FROM Student;
26
27 -- 7. Delete all record of student table
28 DELETE FROM Student;

```

### મેમરી ટ્રીક

“CAIRSCD - Create Add Insert Retrieve Search Count Delete”

## પ્રશ્ન 3(a OR) [3 ગુણ]

SQL માં equi join ઉદાહરણ સાથે સમજાવો.

### જવાબ

**Equi Join:** સમાન કૉલમ્સના આધારે ટેબલ્સને જોડવા માટે સમાનતા શરતનો ઉપયોગ કરતું જોઇન ઓપરેશન.  
ટેબલ:

કોષ્ટક 13. Equi Join

જોઇન પ્રકાર	શરત	પરિણામ
Equi Join	Column1 = Column2	બંને ટેબલ્સમાંથી મેચિંગ રો

```

1 -- ઉદાહરણ
2 SELECT s.name, c.course_name
3 FROM Student s, Course c
4 WHERE s.course_id = c.course_id;

```

- **Operator:** ઇકવાલિટી ઓપરેટર (=) નો ઉપયોગ કરે છે
- **Columns:** સામાન્ય રીતે પ્રાઇમરી કી અને ફોરેન કીની સરખામણી કરે છે

### મેમરી ટ્રીક

“EE - Equi Equals”

## પ્રશ્ન 3(b OR) [4 ગુણ]

નીચેના Aggregate function સમજાવો .1) MAX 2) SUM

### જવાબ

ટેબલ:

કોષ્ટક 14. SQL Aggregate ફંક્શન્સ

ફંક્શન	હેતુ	સિન્ટેક્સ	ઉદાહરણ
MAX	મહત્તમ મૂલ્ય રિટર્ન કરે છે	MAX(column)	MAX(salary)
SUM	કુલ સરવાળો રિટર્ન કરે છે	SUM(column)	SUM(marks)

```

1 SELECT MAX(salary) FROM Employee; -- e.g., 50000
2 SELECT SUM(marks) FROM Student; -- e.g., 450

```

- **Aggregation:** મૂલ્યોના સેટ પર ગણતરી કરીને એક મૂલ્ય આપે છે
- **Usage:** વારંવાર GROUP BY સાથે વપરાય છે

મેમરી ટ્રીક

``MS - MAX Sum``

### પ્રશ્ન 3(c OR) [7 ગુણ]

નીચેના Table માટે SQL સ્ટેટમેન્ટ લખો: Employee(EmpID,Ename,DOB,Dept,Salary)

જવાબ

```

1 -- 1. Create a table Employee
2 CREATE TABLE Employee (
3   EmpID VARCHAR(10) PRIMARY KEY,
4   Ename VARCHAR(50),
5   DOB DATE,
6   Dept VARCHAR(30),
7   Salary DECIMAL(10,2)
8 );
9
10 -- 2. Find sum of salaries of all employee
11 SELECT SUM(Salary) FROM Employee;
12
13 -- 3. Insert one record in Employee table
14 INSERT INTO Employee VALUES
15 ('E001', 'John Doe', '1990-05-15', 'IT', 35000);
16
17 -- 4. Find names of employees who salary between 25000/- and 48000/-
18 SELECT Ename FROM Employee WHERE Salary BETWEEN 25000 AND 48000;
19
20 -- 5. Display detail of all employees in descending order of their DOB
21 SELECT * FROM Employee ORDER BY DOB DESC;
22
23 -- 6. List name of all employees whose name ends with 'a'
24 SELECT Ename FROM Employee WHERE Ename LIKE '%a';
25
26 -- 7. Find highest and least salaries of all employees
27 SELECT MAX(Salary) AS Highest, MIN(Salary) AS Lowest FROM Employee;

```

મેમરી ટ્રીક

``CSIDDHL - Create Sum Insert Display Display List HighLow``

### પ્રશ્ન 4(a) [3 ગુણ]

Normalization ના ફાયદા જણાવો.

## જવાબ

ટેબલ:

કોષ્ટક 15. Normalization લાભો

ફાયદો	વિવરણ
રીડ-ડ-સી ઘટાડો	ડેટા ડુપ્લિકેશન દૂર કરે છે, સ્ટોરેજ બચાવે છે
ડેટા કન્સિસ્ટન્સી	ખાતરી કરે છે કે ડેટા આખા ડેટાબેઝમાં સચોટ રહે
એનોમલી નિવારણ	Insertion, Update અને Deletion એનોમલીઝ અટકાવે છે
ફ્લેક્સિબિલિટી	ડેટા સ્ટ્રક્ચરને ભવિષ્યના ફેરફારો માટે અનુકૂળનક્ષમ બનાવે છે

- **Storage:** ઓછી જગ્યાની જરૂર પડે છે
- **Maintenance:** ડેટા મેનેજમેન્ટ સરળ બનાવે છે
- **Organization:** લોજિકલ ડેટા ગ્રુપિંગ પ્રદાન કરે છે

## મેમરી ટ્રીક

``RCEF - Reducing Consistency Errors Forever``

## પ્રશ્ન 4(b) [4 ગુણ]

1NF અને 2NF સમજાવો.

## જવાબ

**1NF (First Normal Form):**

- કોલમમાં માત્ર એટોમિક (અવિભાજ્ય) મૂલ્યો હોવા જોઈએ.
- કોઈ રીપીટિંગ ગ્રુપ્સ ન હોવા જોઈએ.

**2NF (Second Normal Form):**

- ટેબલ 1NF માં હોવું જોઈએ.
- કોઈ પાર્શિયલ ડિપેન્ડન્સી (Partial Dependency) ન હોવી જોઈએ (Non-prime attributes સંપૂર્ણપણે પ્રાઇમરી કી પર આધારિત હોવા જોઈએ).

ઉદાહરણ:

કોષ્ટક 16. ડિપેન્ડન્સી ઉદાહરણ

Student_ID	Course_ID	Course_Fee
1	C1	5000

અહીં, Course\_Fee માત્ર Course\_ID પર આધારિત છે, જે 2NF નું ઉલ્લંઘન છે.

## મેમરી ટ્રીક

``AtomPart - Atomic values, No Partial dependency``

## પ્રશ્ન 4(c) [7 ગુણ]

નીચેના SQL function સમજાવો. 1)COUNT 2)TO\_CHAR 3)LAST\_DAY 4)ADD\_MONTHS

## જવાબ

ટેબલ:

કોષ્ટક 17. SQL ફંક્શન્સ

ફંક્શન	વિવરણ	ઉદાહરણ
COUNT	રો ની કુલ સંખ્યા ગણે છે	COUNT(*) → 10
TO_CHAR	ડેટાને સ્ટ્રિંગ ફોર્મેટમાં કન્વર્ટ કરે છે	TO_CHAR(SYSDATE, 'MM')
LAST_DAY	મહિનાનો છેલ્લો દિવસ રિટર્ન કરે છે	LAST_DAY(date)
ADD_MONTHS	તારીખમાં મહિના ઉમેરે છે	ADD_MONTHS(date, 2)

```

1 SELECT COUNT(*) FROM EMP;
2 SELECT TO_CHAR(2000, '$9,999') FROM DUAL; -- Returns $2,000
3 SELECT LAST_DAY('01-FEB-2024') FROM DUAL; -- Returns 29-FEB-24
4 SELECT ADD_MONTHS('01-JAN-2024', 1) FROM DUAL; -- Returns 01-FEB-24

```

## મેમરી ટ્રીક

``CTLA - Count To Last Add``

## પ્રશ્ન 4(a OR) [3 ગુણ]

ACID પ્રોપર્ટીઝ સમજાવો.

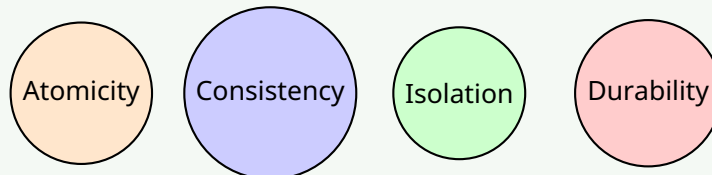
## જવાબ

TRANSACTION: ડેટાબેઝ કામગીરીનું લોજિકલ યુનિટ.

ટેબલ:

કોષ્ટક 18. ACID ગુણધર્મો

પ્રોપર્ટી	વ્યાખ્યા
Atomicity	બધું અથવા કંઈ નહીં (All or Nothing)
Consistency	ટ્રાન્ઝેક્શન પહેલાં અને પછી ડેટા કન્સિસ્ટન્ટ હોવો જોઈએ
Isolation	દરેક ટ્રાન્ઝેક્શન સ્વતંત્ર રીતે એક્ઝિક્યુટ થવું જોઈએ
Durability	કમિટ થયા પછી ફેરફારો કાયમી હોવા જોઈએ



આકૃતિ 7. ACID મોડેલ

## મેમરી ટ્રીક

``ACID - Atomicity Consistency Isolation Durability``

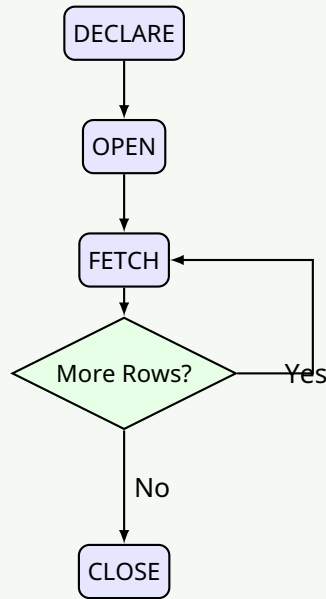
## પ્રશ્ન 4(b OR) [4 ગુણ]

Cursor ના સ્ટેપ્સ લખો.

**જવાબ**

**Explicit Cursor Lifecycle:**

1. **DECLARE:** કર્સરને નામ આપવું અને કવેરી વ્યાખ્યાયિત કરવી (મેમરી ફાળવણી).
  - CURSOR c1 IS SELECT ...;
2. **OPEN:** કવેરી એક્ઝિક્યુટ કરવી અને રિઝલ્ટ સેટ તૈયાર કરવો.
  - OPEN c1;
3. **FETCH:** કર્સરમાંથી ડેટા વન-બાય-વન વેરીએબલ્સમાં લેવો.
  - FETCH c1 INTO v\_name;
4. **CLOSE:** રિસોર્સિસ મુક્ત કરવા.
  - CLOSE c1;



આકૃતિ 8. Cursor Steps

**મેમરી ટ્રીક**

“DOFC - Declare Open Fetch Close”

## પ્રશ્ન 4(c OR) [7 ગુણ]

3NF ઉદાહરણ સાથે સમજાવો.

**જવાબ**

**3NF (Third Normal Form):**

- ટેબલ 2NF માં હોવું જોઈએ.
- કોઈ Transitive Dependency ન હોવી જોઈએ (Non-prime attribute અન્ય non-prime attribute પર આધારિત ન હોવો જોઈએ).

**ઉદાહરણ (Non-3NF):**

Emp_ID	Dept_Name	Dept_Location
101	IT	Block A
102	HR	Block B

અહીં, Dept\_Location એ Dept\_Name પર આધારિત છે, અને Dept\_Name એ Emp\_ID પર. (Transitive: Emp\_ID → Dept\_Name → Dept\_Location).

**ઉકેલ (3NF Conversion):** બે ટેબલમાં વિભાજન કરો: 1. Employee (Emp\_ID, Dept\_Name) 2. Department (Dept\_Name, Dept\_Location)  
ટેબલ:

કોષ્ટક 19. 3NF શરતો

શરત	વિવરણ
2NF	પાર્શિયલ ડિપેન્ડન્સી દૂર થયેલ હોવી જોઈએ
No Transitive	$A \rightarrow B$ અને $B \rightarrow C$ (જ્યાં B non-key છે) ન હોવું જોઈએ

#### મેમરી ટ્રીક

“NoTransitive - No transitive dependencies allowed”

### પ્રશ્ન 5(a) [3 ગુણ]

લોક આધારિત પ્રોટોકોલ સમજાવો.

#### જવાબ

**Lock:** ડેટા આઇટમ સાથે સંકળાયેલ વેરીએબલ જે તે આઇટમ પરના ઓપરેશન્સની સ્થિતિ દર્શાવે છે.

**લોક મિકેનિઝમ:**

- **Shared Lock (S):** રીડ-ઓન્લી લોક. ડેટા વાંચી શકાય છે પણ બદલી શકાતો નથી.
- **Exclusive Lock (X):** રીડ-રાઈટ લોક. ડેટા વાંચી અને બદલી શકાય છે.

ટેબલ:

કોષ્ટક 20. Lock Compatibility Matrix

	Shared (S)	Exclusive (X)
Shared (S)	True (Compatible)	False (Conflict)
Exclusive (X)	False (Conflict)	False (Conflict)

#### મેમરી ટ્રીક

“SX - Shared reads, Exclusive writes”

### પ્રશ્ન 5(b) [4 ગુણ]

ડેટાબેઝ ટ્રિગર સમજાવો.

#### જવાબ

**Trigger:** એક સ્ટોર કરેલ પ્રોસિજર જે કોઈ ચોક્કસ ડેટાબેઝ ઇવેન્ટ (જેમ કે INSERT, UPDATE, DELETE) થાય ત્યારે આપમેળે એક્ઝિક્યુટ થાય છે.

**ઉપયોગો:**

- જટિલ ઇન્ટેગ્રિટી કન્સ્ટ્રેઇન્ટ્સ લાગુ કરવા.
- ઓડિટ ટ્રેલ્સ (Audit Trails) જાળવવા.
- ટેબલ્સ વચ્ચે ડેટા સિંક્રનાઇઝ કરવા.

સિન્ટેક્સ:

```
1 CREATE TRIGGER trigger_name
2 BEFORE/AFTER INSERT/UPDATE/DELETE
3 ON table_name
4 FOR EACH ROW
5 BEGIN
6   -- Logic
7 END;
```

મેમરી ટ્રીક

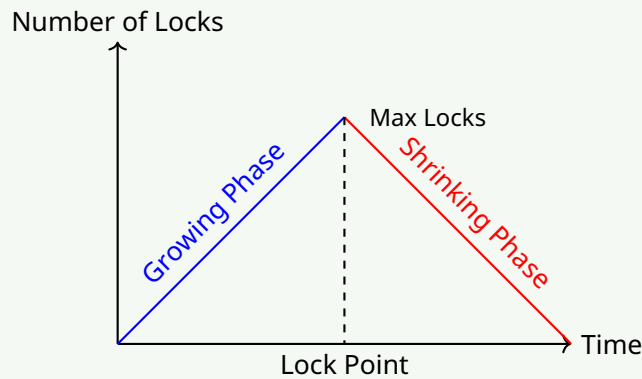
“AutoFire - Automatically fires on events”

## પ્રશ્ન 5(c) [7 ગુણ]

Two Phase Locking (2PL) પ્રોટોકોલ સમજાવો.

જવાબ

**2PL Protocol:** સિરિયલાઇઝેબિલિટીની ખાતરી કરતું કોન્કરન્સી કંટ્રોલ મિકેનિઝમ. ટ્રાન્ઝેક્શનને બે તબક્કામાં વહેંચે છે.



આકૃતિ 9. 2PL Phases

ટેબલ:

કોષ્ટક 21. 2PL તબક્કાઓ

તબક્કો	ક્રિયા
Growing Phase	ટ્રાન્ઝેક્શન લોક્સ મેળવે છે, પણ કોઈ લોક છોડતું નથી.
Shrinking Phase	ટ્રાન્ઝેક્શન લોક્સ છોડે છે, પણ કોઈ નવું લોક મેળવી શકતું નથી.

- Lock Point:** જ્યારે ટ્રાન્ઝેક્શન પાસે મહત્તમ લોક્સ હોય છે.
- Serializability:** ગેરંટી આપે છે કે શીડ્યુલ સિરિયલાઇઝેબલ છે.
- Deadlock:** 2PL માં ડેડલોક થવાની શક્યતા રહે છે.

મેમરી ટ્રીક

“GS - Growing then Shrinking”

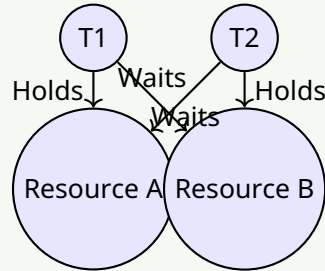
## પ્રશ્ન 5(a OR) [3 ગુણ]

Deadlock એટલે શું?

જવાબ

**Deadlock:** એવી સ્થિતિ જ્યાં બે કે તેથી વધુ ટ્રાન્ઝેક્શન્સ એકબીજા દ્વારા લોક કરેલા રિસોર્સિસની અનંતકાળ સુધી રાહ જુએ છે.  
**ઉદાહરણ:**

- T1 લોક કરે છે A ને, રાહ જુએ છે B ની.
- T2 લોક કરે છે B ને, રાહ જુએ છે A ની.
- સાયકલ બને છે અને કોઈ આગળ વધી શકતું નથી.



આકૃતિ 10. Deadlock Cycle

મેમરી ટ્રીક

"CyclicWait - Cycle of waiting"

## પ્રશ્ન 5(b OR) [4 ગુણ]

Implicit cursor અને Explicit cursor વચ્ચેનો તફાવત આપો.

જવાબ

ટેબલ:

કોષ્ટક 22. Cursor તફાવત

ફીચર	Implicit Cursor	Explicit Cursor
ક્રિએશન	ઓરેકલ દ્વારા આપમેળે બને છે	પ્રોગ્રામર દ્વારા વ્યાખ્યાયિત થાય છે
નામ	'SQL'	યુઝર દ્વારા અપાયેલ નામ
નિયંત્રણ	ઓછું નિયંત્રણ	સંપૂર્ણ નિયંત્રણ (Open, Fetch, Close)
ઉપયોગ	DML (INSERT, UPDATE, DELETE)	SELECT કમાન્ડ જે એકથી વધુ રો આપે છે
એટ્રિબ્યુટ્સ	SQL%FOUND, SQL%ROWCOUNT	c1%FOUND, c1%ROWCOUNT

- **Implicit:** સરળ, ઓછો કોડ, ઓછી ફ્લેક્સિબિલિટી
- **Explicit:** જટિલ, વધુ કોડ, વધારે ફ્લેક્સિબિલિટી

મેમરી ટ્રીક

"AutoVsManual - Oracle handles vs User handles"



## પ્રશ્ન 5(c OR) [7 ગુણ]

Concurrency Control ની સમસ્યાઓની યાદી બનાવો. કોઈપણ બે ઉદાહરણ સાથે સમજાવો.

### જવાબ

Concurrency Control ની સમસ્યાઓ:

1. **Lost Update:** અપૂર્ણ (uncommitted) ડેટાને ઓવરરાઈટ કરવો
  2. **Dirty Read:** અપૂર્ણ (uncommitted) ડેટા વાંચવો
  3. **Non-repeatable Read:** બદલાયેલ ડેટા ફરીથી વાંચવો
  4. **Phantom Read:** રેન્જ ક્વેરીમાં નવી રો (rows) દેખાવી
- ટેબલ:

કોષ્ટક 23. Concurrency સમસ્યાઓ

સમસ્યા	ઉદાહરણ દૃશ્ય
Lost Update	T1 વાંચે છે X, T2 વાંચે છે X. T1 અપડેટ કરે છે X, T2 અપડેટ કરે છે X. T1 નું અપડેટ ખોવાઈ જાય છે.
Dirty Read	T1 અપડેટ કરે છે X. T2 વાંચે છે X. T1 ફેલ/રોલબેક થાય છે. T2 પાસે અમાન્ય ડેટા છે.

**ઉદાહરણ 3:** Lost Update માટે બીજું ઉદાહરણ: T1: Read A(100) → A=A+50 T2: Read A(100) → A=A+30 T1: Write A(150) T2: Write A(130) [T1 ના 150 ને ઓવરરાઈટ કરે છે!] અંતિમ મૂલ્ય 180 હોવું જોઈએ, પરંતુ 130 છે.

**ઉદાહરણ 4 (Dirty Read):** T1: Update A=200 [કમીટ થયેલ નથી] T2: Read A(200) [Dirty] T1: Rollback to 100 T2 200 નો ઉપયોગ કરે છે, જે ક્યારેય સત્તાવાર રીતે અસ્તિત્વમાં નહોતું.

### મેમરી ટ્રીક

“LDNP - Lost Dirty Non-repeatable Phantom”