CS 8492 - DataBase Management System

Unit - 1 : Relational DataBases

Purpose of DataBase system - views of data - data models - database system architecture - Introduction to relational databases - Relational model - Keys - Relational Algebra - SQL fundamentals - Advanced SQL features - Embedded SQL - Dynamic SQL.

Unit - 2 : DataBase Design

Entity Relationship models - ER diagrams - Enhanced ER model - ER ÷ to Relational mapping - Functional dependencies - Non ÷ loss decomposition - First, Second, Third normal forms, dependency reservation - Boyce / codd normal form - Multivalued dependencies and fourth normal forms - joined dependencies and fifth normal forms.

Properties of decomp.
Non - loss decomposition
Dependency preservation

Steps for savepoint:

1. Create
2. Insert
3. Select
4. commit
5. delete
6. savepoint S_1
7. update
8. rollback S_1
9. select
10. savepoint S_2
11. rollback S_2

SQL operators (every changes are temporary) (only when update) after is used, it changes permanently)

1. Arithmetic (+, -, /, *, %)
2. Logical (and, or, not)
3. Comparison (>, <, >=, <=, ==)
4. String (%, like, not like, ||, alias, copy table)
5. set operators (union, intersect, Minus or Except)

Display

Arithmetic

Display the name of the student, reg no = 101, whose marks twice greater than its original marks.

→ will not be changed in database only it displays marks * 2 eg: 70 * 2 = 140

Answer: Select name, Marks * 2 from Student where regno = 101;

Logical

and: Select regno from Student where name = 'priya' and marks = 80;

b) update Bankac set address = 'bbb' where name = 'darsh';

c) delete from Bankac where name = 'darsh';

d) select cusname from Bankac where address = 'ccc';

a> Select cusname from Depositor union select cusname from Borrower;

b> Select cusname from Depositor intersect select cusname from Borrower;

c> Select cusname from Depositor minus select cusname from Borrower;

Referential Integrity :-

Foreign key :

It is a field that points to the primary key of another table.

The purpose of foreign key is to ensure referential integrity of data.

The value appears in one table also appears in another table.

The foreign key is refrencing relation must match the primary key of reference relation.

Syntax :

Create table stock4 (itemno number(5) primary key, itemname varchar2(10), quantity number ; price number)

Create table stock5 (itemno number(5) references stock4 (itemno), brandname varchar2(5));

Insert into stock5 values (5, 'dove');
↳ inserted

Insert into stock5 values (7, 'dove');
↳ not inserted
↳ Error
(we didn't create a primary key as '7' in stock4)

delete from stock4 where itemno = 5;
Error.
↳ first it should be deleted from stock5 and then delete from stock4

Delete child first

(e) Retrieve

Insert into Employee values (101, Moorthy, Ram st, 10000)
(102, Karthick, Rakesh st, 5000)
(103, Sinbu, Kalam st, 11000)

" " Dept " 101, (01, finance dept)
102, (02, research dept)
103, (06, marketing dept)

" " location " (01, Chennai)
(02, Pondicherry)
(06, Vaniyambadi)

Cartesian product (x)

It allows us to combine the informations from two relations.

Syntax :

R x S

R	S
1	3
2	4
3	

R x S	
1	3
1	4
2	3
2	4
3	3
3	4

Join

Natural join (⋈)

eg: Find the name of employee with salary from relation employee (id, name) and empsalary (id, salary)

$\Pi_{empname,\ salary}$ (employee ⋈ empsalary)

Left outer join (⟕)

eg: Find the name of employee with salary from relation employee and empsalary

$\Pi_{empname,\ salary}$ (employee ⟕ empsalary)

Right outer join (⟖)

$\Pi_{empname,\ salary}$ (employee ⟖) empsalary)

Independence :

Ability to modify the schema definition in 1 level without modifying the another level.

Components

> Software -> set of programs
> Hardware -> electronic devices involved
> Data -> collection of raw facts (Meta Data -> Data about the data)
↳ stored in Data Dictionary
> Procedure -> rules & regulations
> Data Access Language -> to communicate with computer

Component modules:

< Query processor
< storage Manager
< Disc storage
< Data Base users and Administrators

Data Models

Structure of a database

Types

ER Model [Entity - Relationship]
Relational Model
Hierarchical model
Network model
Object oriented model
Object Relational model

Object relational model :

Combine the advantages of object oriented model programming with relational databases. Embedded SQL or) static

Dynamic SQL

An SQL statement to be executed are known at the time of coding of program statements. The source code is submitted to SQL pre-compiler which processes the SQL statement

Uses :

Online transaction processing

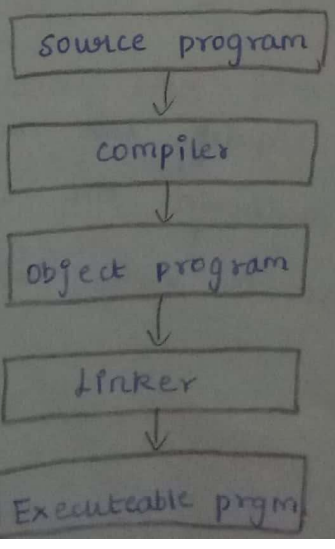
Batch processing

Keywords :

EXEC SQL -> execute SQL Query

END EXEC -> end Query

Life cycle of High level language & Embedded



High level language.

Weak Entity set

It means it may not have a sufficient attribute to form a primary key.

It depends on some other entity.

Eg: payment (payment no, amount) → depends on bankac

Strong Entity set

It means it may have a sufficient attribute to form a primary key.

Eg: Student (Roll no)

ER conditions

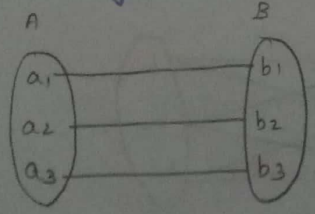
1. Mapping cardinalities
2. Participation constrains — Total, Partial

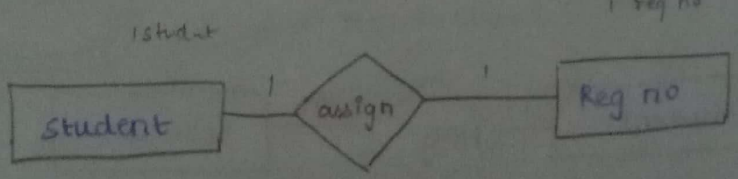
Types of mapping cardinalities

1. one to one
2. one to many
3. many to one
4. Many to many

one to one

An entity A is associated with atmost one entity in B and viceversa.





Reg no ⟶ dno

dno ⟶ dname

Regno ⟶ dname

Normalization:

It is essential part of the database design. It is a process of organizing data in database to avoid redundancy, insertion, updation and deletion anomaly. To overcome these anomalies we need to normalize the data.

Guidelines:

Avoid repeatation of data

Create a table without insertion, deletion and modification anomalies.

Avoid mixture of multiple entities from multiple relation.

Avoid null values.

First Normal Form:

As per the rule of ~~one enough~~ first normal form, an attribute of table cannot hold multiple values.

It holds only atomic values (single)