Important Lectures

Font Color	Importance Level
RED*	VERY HIGH
RED	HIGH
GREEN	MEDIUM
BALCK	LOW

- Lecture 02 -03
- Lecture 04-07 + 07 Practice*
- Lecture 08-11 + 11 Practice*
- SQL From the Lab Lectures*
- Lecture 12
- Lecture 13-16*
- Lecture 17-20*

- Lecture 21
- Lecture 22-26
- Lecture 27 - 28
- Lecture 29- 31
- Lecture 32
- Lecture 33-34

Quiz 01: Lecture 02 - Lecture 07

Quiz 02: Lecture 08 - Lecture 11

Quiz 03: Lecture 12 - Lecture 20

Quiz 04: Lecture 21 - Lecture 26

B Tree: https://www.cs.usfca.edu/~galles/visualization/BTree.html

B+ Tree: https://www.cs.usfca.edu/~galles/visualization/BPlusTree.html

Solution: https://www.db-book.com/db6/practice-exer-dir/index.html

Relational Algebra and Query practice

https://www.exploredatabase.com/2017/10/solved-exercises-in-dbms-answers-explained.html

Let's look at each of the keys in DBMS with example: https://www.guru99.com/dbms-keys.html

Super Key – A super key is a group of single or multiple keys which identifies rows in a table.

Primary Key – is a column or group of columns in a table that uniquely identify every row in that table.

Candidate Key – is a set of attributes that uniquely identify tuples in a table. Candidate Key is a super key with no repeated attributes.

Alternate Key – is a column or group of columns in a table that uniquely identify every row in that table.

Foreign Key – is a column that creates a relationship between two tables. The purpose of foreign keys is to maintain data integrity and allow navigation between two different instances of an entity.

Composite Key – is a combination of two or more columns that uniquely identify rows in a table. The combination of columns guarantees uniqueness, though individual uniqueness is not guaranteed.

Compound Key – has two or more attributes that allow you to uniquely recognize a specific record. It is possible that each column may not be unique by itself within the database.

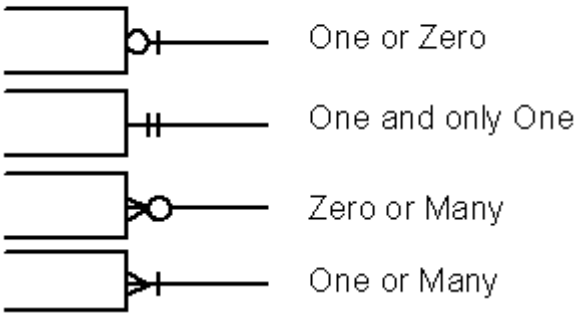
Surrogate Key – An artificial key which aims to uniquely identify each record is called a surrogate key. These kinds of key are unique because they are created when you don't have any natural primary key.

The SQL language has several parts:

• Data-definition language (DDL). The SQL DDL provides commands for defining relation schemas, deleting relations, and modifying relation schemas.

• Data-manipulation language (DML). The SQL DML provides the ability to query information from the database and to insert tuples into, delete tuples from, and modify tuples in the database

Summary of Crow's Foot Notation



Relational algebra – solved exercise

Relational algebra operators:

σ – selection with conditions (It selects all tuples that satisfies the conditions. Shows entire table with respect to the structure)

Π – projection operator (It selects the attributes which are listed here)

⋈ - natural join operator (Binary operator that join two relations on common attributes' values)

-, ∪, and ∩ - set operators (difference, union and intersection)

Question:

Consider the following relational database schema consisting of the four relation schemas:

passenger (pid, pname, pgender, pcity)

agency (aid, aname, acity)

flight (fid, fdate, time, src, dest)

booking (pid, aid, fid, fdate)

Answer the following questions using relational algebra queries;

Solution:

Most of the following queries can be written in many different ways.

a) Get the complete details of all flights to New Delhi.

$\sigma_{\text{destination = "New Delhi"}}$ (flight)

b) Get the details about all flights from Chennai to New Delhi.

$\sigma_{\text{src = "Chennai" ^ dest = "New Delhi"}}$ (flight)

c) Find only the flight numbers for passenger with pid 123 for flights to Chennai before 06/11/2020.

Π_{fid} ($\sigma_{\text{pid = 123}}$ (booking) $\bowtie$ $\sigma_{\text{dest = "Chennai" ^ fdate < 06/11/2020}}$ (flight))

[Hint: Given conditions are pid, dest, and fdate. To get the flight id for a passenger given a pid, we have two tables flight and booking to be joined with necessary conditions. From the result, the flight id can be projected]

d) Find the passenger names for passengers who have bookings on at least one flight.

Π_{pname} (passenger $\bowtie$ booking)

e) Find the passenger names for those who do not have any bookings in any flights.

Π_{pname} ((Π_{pid} (passenger) - Π_{pid} (booking)) $\bowtie$ passenger)

[Hint: here applied a set difference operation. The set difference operation returns only pids that have no booking. The result is joined with passenger table to get the passenger names.]

f) Find the agency names for agencies that located in the same city as passenger with passenger id 123.

$\Pi_{aname}(agency \bowtie_{acity = pcity} (\sigma_{pid = 123}(passenger)))$

[Hint: we performed a theta join on equality conditions (equi join) here. This is done between details of passenger 123 and the agency table to get the valid records where the city values are same. From the results, aname is projected.]

g) Get the details of flights that are scheduled on both dates 01/12/2020 and 02/12/2020 at 16:00 hours.

$(\sigma_{fdate = 01/12/2020 \wedge time = 16:00}(flight)) \cap (\sigma_{fdate = 02/12/2020 \wedge time = 16:00}(flight))$

[Hint: the requirement is for flight details for both dates in common. Hence, set intersection is used between the temporary relations generated from application of various conditions.]

h) Get the details of flights that are scheduled on either of the dates 01/12/2020 or 02/12/2020 or both at 16:00 hours.

$(\sigma_{fdate = 01/12/2020 \wedge time = 16:00}(flight)) \cup (\sigma_{fdate = 02/12/2020 \wedge time = 16:00}(flight))$

i) Find the agency names for agencies who do not have any bookings for passenger with id 123.

$\Pi_{aname}(agency \bowtie (\Pi_{aid}(agency) - \Pi_{aid}(\sigma_{pid = 123}(booking)))$

j) Find the details of all male passengers who are associated with Jet agency.

$\Pi_{passengers.pid,\ pname,\ pcity}(\sigma_{pgender\ =\ \text{“Male”}}(passengers \bowtie booking \bowtie agency))$

[Hint: To get the link between passengers and agency, we need to join all three tables passengers, booking, and agency with necessary condition. Here, agency links both passengers and agency. As we have performed natural join operation between all three tables, the degree of the result will consist of all attributes from all the three tables. Hence, we project only passengers details as these are mentioned as required.]

Basic Steps in Query Processing

1. Parsing and translation
2. Optimization
3. Evaluation

1. **Parsing and translation**
2. Translate the query into its internal form. This is then translated into relational algebra.
3. Parser checks syntax, verifies relation.
4. **Optimization**
5. SQL is a very high level language:

6. The users specify what to search for- not how the search is actually done
7. The algorithms are chosen automatically by the DBMS.

- For a given SQL query there may be many possible execution plans.
- Amongst all equivalent plans choose the one with lowest cost.
- Cost is estimated using statistical information from the database catalog.
- Evaluation
- The query evaluation engine takes a query evaluation plan, executes that plan and returns the answer to that query.

With this information, let us write a query to find the list of all employees who are working in a project which is more than 10 months old.

```
SELECT Ename
FROM Employee, Proj_Assigned
WHERE Employee.Eno = Proj_Assigned.Eno AND DOP > 10;
```

Input:

A query written in SQL is given as input to the query processor. For our case, let us consider the SQL query written above.

Step 1: Parsing

In this step, the parser of the query processor module checks the syntax of the query, the user's privileges to execute the query, the table names and attribute names, etc. The correct table names, attribute names and the privilege of the users can be taken from the system catalog (data dictionary).

Step 2: Translation

If we have written a valid query, then it is converted from high level language SQL to low level instruction in Relational Algebra.

For example, our SQL query can be converted into a Relational Algebra equivalent as follows;

$\pi_{Ename}(\sigma_{DOP>10 \wedge Employee.Eno=Proj\_Assigned.Eno}(Employee \times Prof\_Assigned))$

Step 3: Optimizer

Optimizer uses the statistical data stored as part of data dictionary. The statistical data are information about the size of the table, the length of records, the indexes created on the table, etc. Optimizer also checks for the conditions and conditional attributes which are parts of the query.

Step 4: Execution Plan

A query can be expressed in many ways. The query processor module, at this stage, using the information collected in step 3 to find different relational algebra expressions that are equivalent and return the result of the one which we have written already.

For our example, the query written in Relational algebra can also be written as the one given below;

$\pi_{Ename}(Employee \bowtie_{Eno} (\sigma_{DOP>10}(Prof\_Assigned)))$

So far, we have got two execution plans. Only condition is that both plans should give the same result.

Step 5: Evaluation

Though we got many execution plans constructed through statistical data, though they return same result (obvious), they differ in terms of Time consumption to execute the query, or the Space required executing the query. Hence, it is mandatory choose one plan which obviously consumes less cost.

At this stage, we choose one execution plan of the several we have developed. This Execution plan accesses data from the database to give the final result.

In our example, the second plan may be good. In the first plan, we join two relations (costly operation) then apply the condition (conditions are considered as filters) on the joined relation. This consumes more time as well as space.

In the second plan, we filter one of the tables (Proj_Assigned) and the result is joined with the Employee table. This join may need to compare less number of records. Hence, the second plan is the best (with the information known, not always).

Output:

The final result is shown to the user.

ACID properties of transactions

In the context of transaction processing, the acronym *ACID* refers to the four key properties of a transaction: atomicity, consistency, isolation, and durability.

Atomicity

All changes to data are performed as if they are a single operation. That is, all the changes are performed, or none of them are.

For example, in an application that transfers funds from one account to another, the atomicity property ensures that, if a debit is made successfully from one account, the corresponding credit is made to the other account.

Consistency

Data is in a consistent state when a transaction starts and when it ends.

For example, in an application that transfers funds from one account to another, the consistency property ensures that the total value of funds in both the accounts is the same at the start and end of each transaction.

Isolation

The intermediate state of a transaction is invisible to other transactions. As a result, transactions that run concurrently appear to be serialized.

For example, in an application that transfers funds from one account to another, the isolation property ensures that another transaction sees the transferred funds in one account or the other, but not in both, nor in neither.

Durability

After a transaction successfully completes, changes to data persist and are not undone, even in the event of a system failure.

For example, in an application that transfers funds from one account to another, the durability property ensures that the changes made to each account will not be reversed.

Link: ACID properties of transactions - IBM Documentation

Conflict Serializability in DBMS - GeeksforGeeks

Precedence Graph For Testing Conflict Serializability in DBMS - GeeksforGeeks

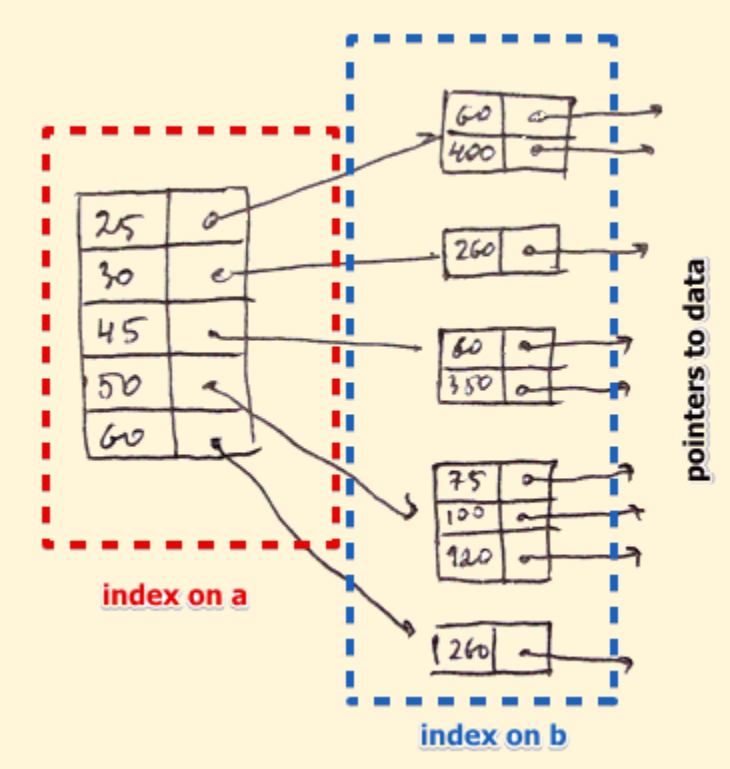
Conflict Serializable: A schedule is called conflict serializable if it can be transformed into a serial schedule by swapping non-conflicting operations.
Conflicting operations: Two operations are said to be conflicting if all conditions satisfy:

- They belong to different transactions
- They operate on the same data item
- At Least one of them is a write operation

Conflict Equivalent: Two schedules are said to be conflict equivalent when one can be transformed to another by swapping non-conflicting operations.

What is Multilevel Index?

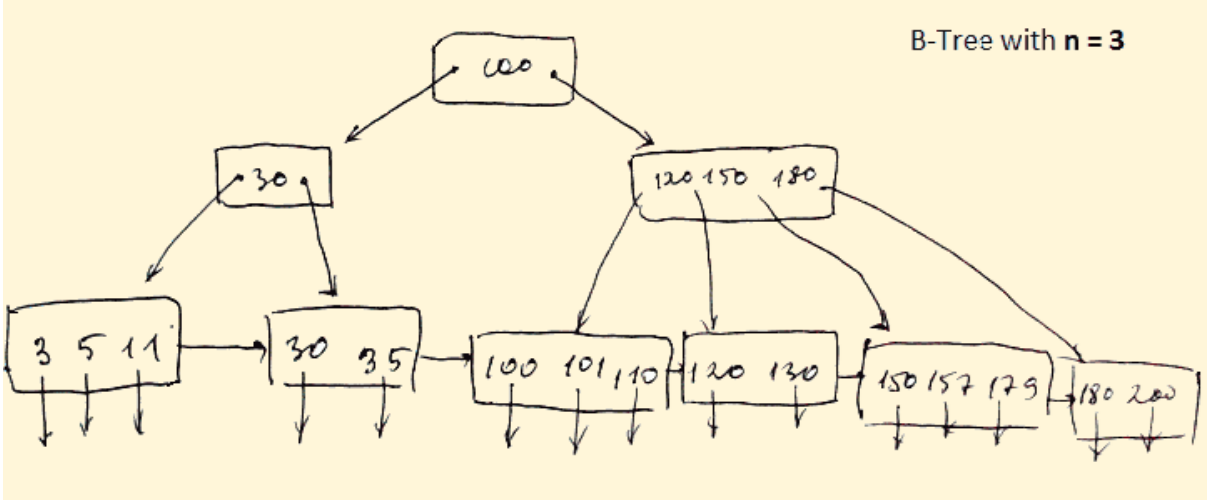
Multilevel Indexing in Database is created when a primary index does not fit in memory. In this type of indexing method, you can reduce the number of disk accesses to short any record and kept on a disk as a sequential file and create a sparse base on that file.



B-Tree Index

B-tree index is the widely used data structures for tree based indexing in DBMS. It is a multilevel format of tree based indexing in DBMS technique which has balanced binary search trees. All leaf nodes of the B tree signify actual data pointers.

Moreover, all leaf nodes are interlinked with a link list, which allows a B tree to support both random and sequential access.



Lead nodes must have between 2 and 4 values.

Every path from the root to leaf are mostly on an equal length.

Non-leaf nodes apart from the root node have between 3 and 5 children nodes.

Every node which is not a root or a leaf has between n/2] and n children.

Advantages of Indexing

Important pros/ advantage of Indexing are:

It helps you to reduce the total number of I/O operations needed to retrieve that data, so you don't need to access a row in the database from an index structure.

Offers Faster search and retrieval of data to users.

Indexing also helps you to reduce tablespace as you don't need to link to a row in a table, as there is no need to store the ROWID in the Index. Thus you will able to reduce the tablespace.

You can't sort data in the lead nodes as the value of the primary key classifies it.

Disadvantages of Indexing

Important drawbacks/cons of Indexing are:

To perform the indexing database management system, you need a primary key on the table with a unique value.

You can't perform any other indexes in Database on the Indexed data.

You are not allowed to partition an index-organized table.

SQL Indexing Decrease performance in INSERT, DELETE, and UPDATE query.

Summary:

Indexing is a small table which is consist of two columns.

Two main types of indexing methods are 1)Primary Indexing 2) Secondary Indexing.

Primary Index is an ordered file which is fixed length size with two fields.

The primary Indexing is also further divided into two types 1)Dense Index 2)Sparse Index.

In a dense index, a record is created for every search key valued in the database.

A sparse indexing method helps you to resolve the issues of dense Indexing.

The secondary Index in DBMS is an indexing method whose search key specifies an order different from the sequential order of the file.

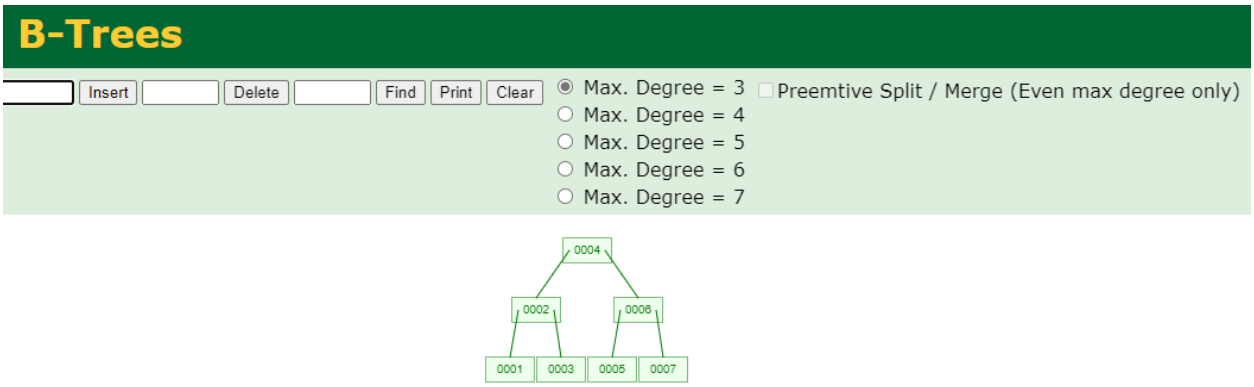
Clustering index is defined as an order data file.

Multilevel Indexing is created when a primary index does not fit in memory.

The biggest benefit of Indexing is that it helps you to reduce the total number of I/O operations needed to retrieve that data.

The biggest drawback to performing the indexing database management system, you need a primary key on the table with a unique value.

B Tree

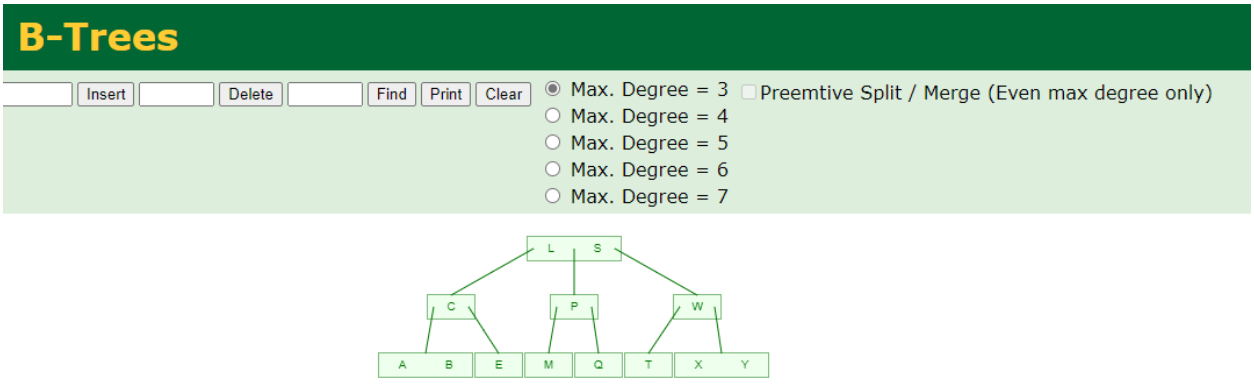


Practice Problem

- Assume a tree where each node can contain three pointers . Enter the values

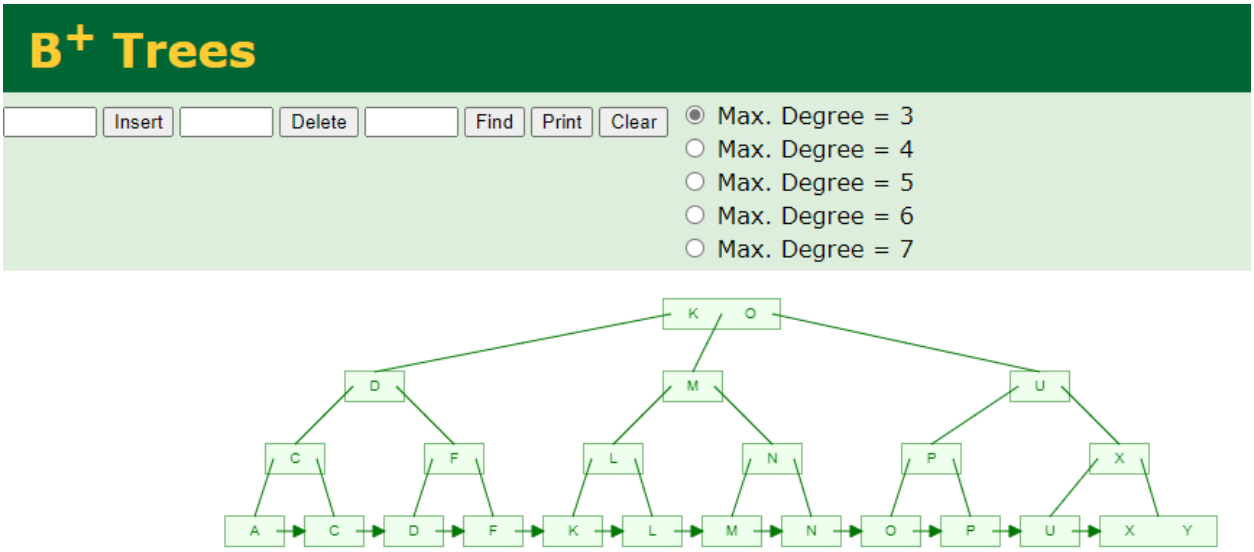
S W A E C T M L P Y X Q B

*Solve Using B Tree



B+ Tree

Sorted Order: A C D F K L M N O P U X Y



Fragmentation in Distributed DBMS

Fragmentation is a process of dividing the whole or full database into various subtables or sub relations so that data can be stored in different systems. The small pieces of sub relations or subtables are called fragments

In the fragmentation process, let's say, If a table T is fragmented and is divided into a number of fragments say T1, T2, T3….TN.

Advantages :

As the data is stored close to the usage site, the efficiency of the database system will increase

Local query optimization methods are sufficient for some queries as the data is available locally

In order to maintain the security and privacy of the database system, fragmentation is advantageous

Disadvantages :

Access speeds may be very high if data from different fragments are needed

If we are using recursive fragmentation, then it will be very expensive

We have three methods for data fragmenting of a table:

Horizontal fragmentation

Vertical fragmentation

Mixed or Hybrid fragmentation

Let's discuss them one by one.

Horizontal fragmentation –

Horizontal fragmentation refers to the process of dividing a table horizontally by assigning each row or (a group of rows) of relation to one or more fragments. These fragments are then be assigned to different sides in the distributed system. Some of the rows or tuples of the table are placed in one system and the rest are placed in other systems.

For example, consider an EMPLOYEE table (T) :

Eno	Ename	Design	Salary	Dep
101	A	abc	3000	1
102	B	abc	4000	1
103	C	abc	5500	2
104	D	abc	5000	2

105	E	abc	2000	2

This EMPLOYEE table can be divided into different fragments like:

EMP 1 = σDep = 1 EMPLOYEE

EMP 2 = σDep = 2 EMPLOYEE

These two fragments are: T1 fragment of Dep = 1

Eno	Ename	Design	Salary	Dep
101	A	abc	3000	1
102	B	abc	4000	1

Similarly, the T2 fragment on the basis of Dep = 2 will be :

Eno	Ename	Design	Salary	Dep
103	C	abc	5500	2
104	D	abc	5000	2
105	E	abc	2000	2

Now, here it is possible to get back T as T = T1 ∪ T2 ∪ …. ∪ TN

Vertical Fragmentation

Vertical fragmentation refers to the process of decomposing a table vertically by attributes are columns. In this fragmentation, some of the attributes are stored in one system and the rest are stored in other systems. This is because each site may not need all columns of a table. In order to take care of restoration, each fragment must contain the primary key field(s) in a table.

For example, for the EMPLOYEE table we have T1 as :

Eno	Ename	Design	Tuple_id
101	A	abc	1
102	B	abc	2
103	C	abc	3
104	D	abc	4
105	E	abc	5

For the second. sub table of relation after vertical fragmentation is given as follows :

Salary	Dep	Tuple_id
3000	1	1
4000	2	2
5500	3	3
5000	1	4
2000	4	5

This is T2 and to get back to the original T, we join these two fragments T1 and T2 as πEMPLOYEE (T1 ⋈ T2)

https://www.geeksforgeeks.org/fragmentation-in-distributed-dbms/

difference between Star and Snowflake Schema:

S.NO	Star Schema	Snowflake Schema
1.	In star schema, The fact tables and the dimension tables are contained.	While in snowflake schema, The fact tables, dimension tables as well as sub dimension tables are contained.
2.	Star schema is a top-down model.	While it is a bottom-up model.
3.	Star schema uses more space.	While it uses less space.
4.	It takes less time for the execution of queries.	While it takes more time than star schema for the execution of queries.
5.	In star schema, Normalization is not used.	While in this, Both normalization and denormalization are used.
6.	It's design is very simple.	While it's design is complex.
7.	The query complexity of star schema is low.	While the query complexity of snowflake schema is higher than star schema.
8.	It's understanding is very simple.	While it's understanding is difficult.
9.	It has less number of foreign keys.	While it has more number of foreign keys.

10.	It has high data redundancy.	While it has low data redundancy.

Ahsanullah University of Science & Technology

Department of Computer Science & Engineering

Subject: Database System (CSE 3103)

Time: 40 min **Quiz: 01** **Marks: 20**

Name: **ID:**

Marks Obtained

Instructions:

1. *Fill up your Name, ID in the upper portion of the Examination Script.*
2. *Copying, Side talking & seeing is strictly prohibited.*
3. *Use 2 minutes for each marks to answer. No extra time will be given.*

1. Now a days we are using databases in our day to day life. Give five examples with a short description of databases from your regular life where you are using databases. Ignore the common ones like mobile phone contacts/file browser, social networks, mailing services, university results and departmental stores.

2. This is a common scene of a University Library. Borrowers take books out on loan. A borrower may take out up to five books at any one time. Several copies of the same book are held for books which are continually in demand. Borrowers may make reservations for titles which are out on loan. Items which need to be stored include borrower number, borrower name, date borrowed, ISBN, acquisition number (allocated when a book is purchased), date acquired, title, author, borrower making reservation, and date reserved.

The library needs to be able to do the following:

- check whether a book returned has been reserved
- check which borrower has a particular book out on loan
- check which books a borrower has out on loan
- check whether a copy of a particular title is in the library

Draw an ERD to represent the library. Resolve the many-to-many relationships and suggest identifiers and the main attributes for each entity.

3. A student accommodation service looks after several residential halls, and wishes to integrate management of these into a single database. One small part of this concerns recording the allocation of students to rooms. You have the following information, arising from an initial requirements analysis.

- Each student has a universal username (UUN) that can be used to identify them in the database. This is a unique 8-character alphanumeric code issued by the University.
- Other than that, the database must record every student's name and their year of study.
- Students are allocated to rooms. Each room has a fixed capacity — one, two, three, or even more.
- No student can be allocated to more than one room; but some students in the system might not yet have any room assigned.
- There are several halls, each with its own name and unique Building ID.
- Each room belongs to one hall, and is identified within that hall by its number. There is no coordination in room numbering between different halls: in particular, two rooms in different halls might have the same number.

Draw an entity-relationship (ER) diagram that represents this information. Make sure that you capture the constraints described on the relationships involved, and designate appropriate primary keys for all entities.

Department of Computer Science & Engineering

Subject: Database System (CSE 3103)

Time: 30 min | **Quiz: 02 (SET-B)** | **Marks: 20**

Name: ID:

Marks Obtained

10

Instructions:

1. *Fill up your Name, ID in the upper portion of the Examination Script.*
2. *Copying, Side talking & seeing is strictly prohibited.*
3. *Use 2 minutes for each marks to answer. No extra time will be given.*

1. ***Maker(maker_id, maker_name, maker_city)***
   pc(model_id, maker_id, speed, ram, hd, price)
   laptop(model_id, maker_id, speed, ram, hd, screen, price)

   Consider the Computer_Info Database above, where the primary keys are underlined. Write the following queries in relational algebra.

a. **Find the model id and maker id of the lowest priced laptop.**

b. **Decrease the price of laptops by 10% if the current price is less than 100000. Otherwise, decrease the price by 5%.**

c. **For all manufacturers, find the total number of PCs they produce. Display the result along with the manufacturer information (id and name).**

d. **Delete PCs that have manufactured by 'HP'.**

e. **Find the id, name and city of manufacturer(s) with the fastest processor among all those PCs that have the smallest amount of RAM.**

Department of Computer Science & Engineering
Subject: Database System (CSE 3103)
Quiz: 02 (SET-B)

Time: 30 min

Marks: 20

Name: ID:

Instructions:

1. *Fill up your Name, ID in the upper portion of the Examination Script.*
2. *Copying, Side talking & seeing is strictly prohibited.*
3. *Use 2 minutes for each marks to answer. No extra time will be given.*

Marks Obtained

10

1. ***Maker(maker_id, maker_name, maker_city)***
   pc(model_id, maker_id, speed, ram, hd, price)
   laptop(model_id, maker_id, speed, ram, hd, screen, price)

Consider the Computer_Info Database above, where the primary keys are underlined. Write the following queries in relational algebra.

a. **Find the model id and maker id of the lowest priced laptop.**

$\Pi_{model\_id,\ maker\_id}\left(\mathcal{G}_{MIN(price)}(laptop)\right)$

b. **Decrease the price of laptops by 10% if the current price is less than 100000. Otherwise, decrease the price by 5%.**

$n.price \rightarrow price \quad n.laptop \rightarrow laptop$

$\left(\sigma_{n.price > 100000}\right) n.price = n.price - (n.price * 1.1)\ (laptop)$

$\wedge \left(\sigma_{n.price <= 100000}\right) n.price = n.price - (n.price * 1.05)\ (laptop)$

c. **For all manufacturers, find the total number of PCs they produce. Display the result along with the manufacturer information (id and name).**

$\Pi_{maker\_id,\ maker\_name}\left(\mathcal{G}_{count(model\_id)}(maker \bowtie PC)\right)$

d. **Delete PCs that have manufactured by 'HP'.**

$\Pi_{model\_id}(PC) - \Pi_{model\_id}\left(\sigma_{maker\_name = "HP"}(maker \bowtie PC)\right)$

e. **Find the id, name and city of manufacturer(s) with the fastest processor among all those PCs that have the smallest amount of RAM.**

$\Pi_{maker\_id,\ maker\_name,\ maker\_city}\left(\mathcal{G}_{max(speed),\ min(ram)}(PC \bowtie maker)\right)$

Ahsanullah University of Science & Technology

Department of Computer Science & Engineering

Subject: Database System (CSE 3103)

Time: 30 min | Quiz: 02 (SET-B) | Marks: 20

10

2. *Movie (mID, title, year, director)*
   Reviewer (rID, name)
   Rating (rID, mID, stars, ratingDate)

Consider the Movie database above, where the primary keys are underlined. Write the following queries in relational algebra.

f. **Find the titles of all movies directed by James Cameron.**

$\Pi_{title}(\sigma_{director = \text{"James Cameron"}}(movie))$

g. **Find the names of all reviewers who have contributed four or more ratings.**

$\Pi_{name}(\sigma_{stars >= 4}(Reviewer \bowtie Rating))$

h. **Find all years that have a movie that received a rating of 3 or 5.**

$\Pi_{years}(\sigma_{stars = 3 \vee stars = 5}(movie \bowtie Rating))$

i. **Find title, release year and director of the movie that received the higher average stars.**

$\Pi_{title, year, director}$ ~~$\mathcal{G}_{avg(stars)}$~~ $(\sigma_{stars > \mathcal{G}_{avg(stars)}}(Rating \bowtie Movie))$

j. **Some reviewers didn't provide a date with their rating. Find the names of all reviewers who have ratings with a NULL value for the date.**

$\Pi_{names}(\sigma_{rating\_date = \text{"NULL"} \vee ratingDate = \text{" "}}(reviewer \bowtie Rating))$

Set A

Ahsanullah University of Science & Technology

Department of Computer Science & Engineering

Subject: Database System (CSE 3103)

Time: 40 min Quiz: 03 **Marks: 20**

Name: ID:

Marks Obtained

Instructions:

1. *Fill up your Name, ID in the upper portion of the Examination Script.*
2. *Copying, Side talking & seeing is strictly prohibited.*
3. *Use 2 minutes for each marks to answer. No extra time will be given.*

1. A database schema named "PROJECT" contains the following relations which are already in 1NF. You have to decompose the following relations in such a way so that the resulting relations are in 3NF. Explain your answer at each step and identify the primary keys as necessary. A sample dataset is given below for these two relations. 10

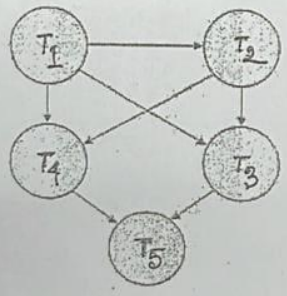
Project

Project Code	Project_Title	Project_Manager	Project_Budget
PC010	Pension System	M Phillips	24500
PC045	Salary System	H Martin	17400
PC064	HR System	K Lewis	12250

Project_Employee_Department

Project Code	Employee No	Employee_Name	Dept_No	Dept_Name	Hourly_Rate
PC010	S10001	A Smith	L004	IT	22.00
PC010	S10030	L Jones	L023	Pension	18.50
PC010	S21010	P Lewis	L004	IT	21.00
PC045	S10010	B Jones	L004	IT	21.75
PC045	S10001	A Smith	L004	IT	18.00
PC045	S31002	T Gilbert	L028	Database	25.00
PC045	S13210	W Richard	L008	Salary	17.00
PC064	S31002	T Gilbert	L028	Database	23.25
PC064	S21010	P Lewis	L004	IT	17.50
PC064	S10034	B James	L009	HR	16.50

2. Consider the precedence graph of Figure below. Is the corresponding schedule conflict serializable? Explain your answer. 5



3. Consider a database for an airline where the database system uses snapshot isolation. Describe a particular scenario in which a nonserializable execution occurs, but the airline may be willing to accept it in order to gain better overall performance. 5

START WRITING FROM NEXT PAGE

Ques 1 Solution

Dept_detail

Dept_No	Dept_Name
L004	IT
L023	Pension

This table is in 1Nf cause values in each column are ~~tot~~ atomic and of same domain and have unique column name. Also in 2Nf as in 1Nf and ~~Employee~~ department name fully functional dependent on dept_no. It is also in 3Nf as No transitivity.

Employee details

Employee_No	Employe_Name
S10001	smith
S10010	James

Same description

Project_details

Project_No	Dept_No	Emp_No	Hour_rate
PC010	L004	S10001	22.00
PC010	L023	S10010	18.50

Same description

Ahsanullah University of Science & Technology

Department of Computer Science & Engineering

Subject: Database System (CSE 3103)

Time: 20 min **Quiz: 04** **Marks: 20**

Name: **ID:**

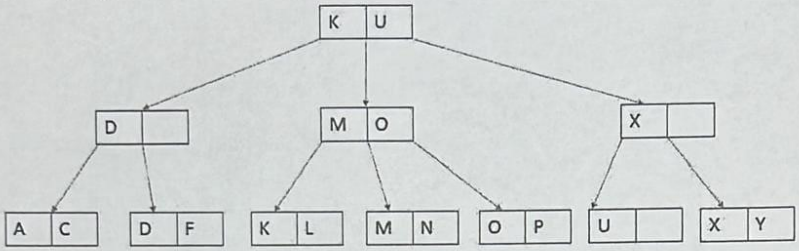
Marks Obtained

Instructions:

1. *Fill up your Name, ID in the upper portion of the Examination Script.*
2. *Copying, Side talking & seeing is strictly prohibited.*
3. *Use 2 minutes for each marks to answer. No extra time will be given.*

1. Consider the following B+ tree where branching factor m=3. Perform the following operation separately and show the tree. **10**

a. Insert : S , Q , G
b. Delete: O , K , M



2. Construct a B tree with the following values where branching factor m=3. You have to show each of the step. **10**

a. Insert : Q , B , H , U , D , S , W , A , E , C , T , M , L , P , Y , X

START WRITING FROM HERE