APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY
FOURTH SEMESTER BTECH DEGREE EXAMINATION, JULY 2017
Course Code: CS208
SUPPLYMENTRY EXAM
Course Name: PRINCIPLES OF DATABASE DESIGN (CS,IT)

Answer key & Scheme of valuation

PART A

(Answer all questions. Each carries 3 marks)

1. *What are the responsibilities of the DBA?* *(3 Marks)*

List any three responsibilities of the DBA. **1mark for 1 responsibility**

The DBA is responsible for:

(i) Designing the logical and physical schemas, as well as widely used portions of the external schema.

(ii) Security and authorization.

(iii) Data availability and recovery from failures.

(iv) Database tuning: The DBA is responsible for evolving the database, in particular the conceptual and physical schemas, to ensure adequate performance as user requirements change.

(v) Acquiring software and hardware resources as needed.

2. *Define the following terms:* *(3 Marks)*

i) data model ii) database schema iii) meta-data

(i) **data model** :

A data model is a collection of conceptual tools for describing data, data relationships, data semantics and consistency constraints. (1 Mark)

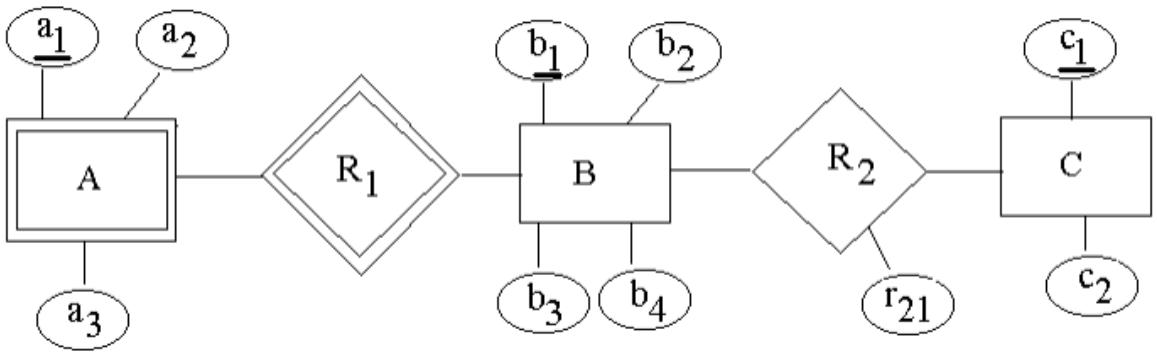
(ii) **database Schema**:

The description of a database is called the database schema, which is specified during database design and is not expected to change frequently. (1 Mark)

(iii) **meta-data**:

Meta-data is the data about data. The schema of a table is an example of meta-data. (1 Mark)

3. *Consider the following ER diagram. Using this ER diagram create a relational database. (Primary keys are underlined)* *(3 marks)*



The relational database schema for the given ER diagram is as follows:

A(a1,b1,a2,a3)

B(b1,b2,b3,b4)

C(c1,c2)

D(b1,c1,r21)

Full marks to this question can be awarded if all the above 4 relations written correctly along with their attributes. Primary keys should be underlined. It is not necessary to the name of the last relation is D. But the attributes of the relation must be b1, c1, and r21.

4. *What are the different ways of classifying a DBMS?*

The DBMS can be classified

Based on Data model:

- Relational
- Object
- Hierarchical and network (legacy)
- Native XML DBMS

Based on Number of users:

- Single-user
- Multiuser

Based on Number of sites:

- Centralized
- Distributed
  - (i) Homogeneous
  - (ii)Heterogeneous

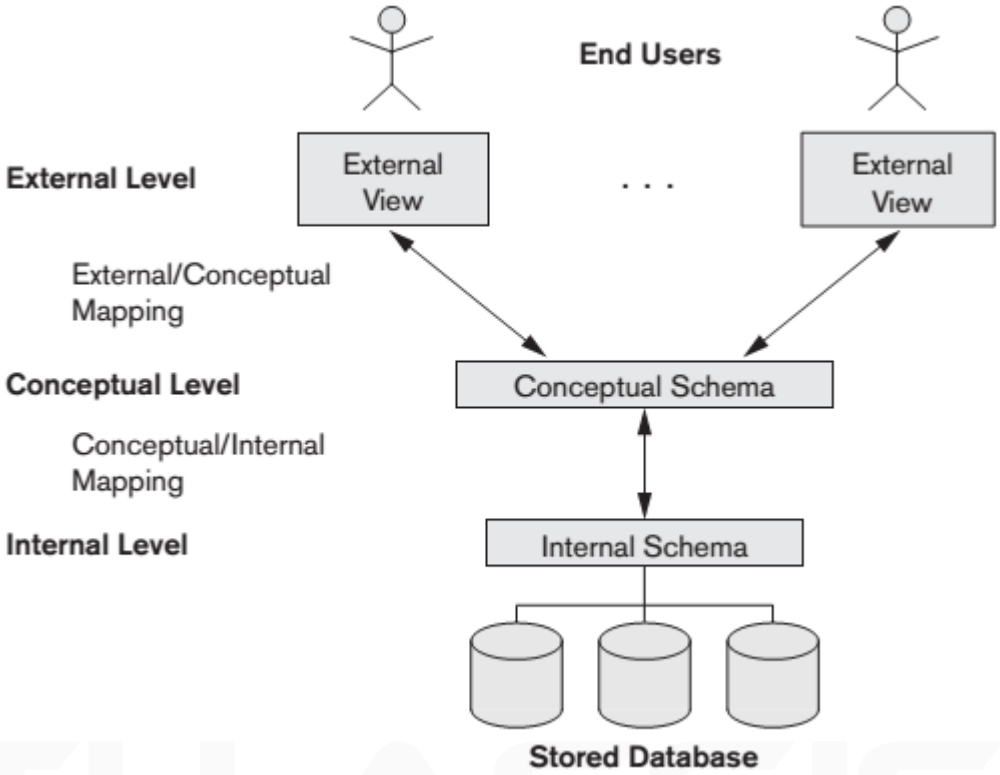
Based on Cost

- Open source
- Different types of licensing

PART B

(Answer any two questions. Each carries 9 marks)

5. *With the help of a neat diagram explain the three schema architecture of DBMS.* *(9 Marks)*



(3 Marks- above figure)

(i) The **internal level** has an internal schema, which describes the physical storage structure of the database. The internal schema uses a physical data model and describes the complete details of data storage and access paths for the database. (2 Marks)

(ii) The **conceptual level** has a conceptual schema, which describes the structure of the whole database for a community of users. The conceptual schema hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints. Usually, a representational data model is used to describe the conceptual schema when a database system is implemented. This implementation conceptual schema is often based on a conceptual schema design in a high-level data model. (2 Marks)

(iii) The **external or view level** includes a number of external schemas or user views. Each external schema describes the part of the database that a particular user group is interested in and hides the rest of the database from that user group. As in the previous level, each external schema is typically implemented using a representational data model, possibly based on an external schema design in a high-level data model. (2 Marks)

6. *Explain the following terms briefly:* *(9 marks)*

(i) Participation constraint
(ii) Overlap constraint
(iii) Covering constraint

(i) **Participation constraint**:

The participation Constraints shows whether the existence of an entity depends on its being associated to another entity by the relationship type. There are two kinds of participation constraints: (1 Mark)

Total: When the entire entities from an entity set participate in a relationship type, is known as total participation, for instance, the participation of the entity set student in the relationship set have to 'opts' is said to be total as each student enrolled must opt for a course. (1 Mark)

Partial: When it is not necessary for all the entities from an entity set to partake in a relationship type, it is known as partial participation. For instance, the participation of the entity set student in 'represents' is partial, as not every student in a class is a class representative. (1 Mark)

(ii) **Overlap constraint**:

Within an ISA hierarchy, an overlap constraint determines whether or not two subclasses can contain the same entity. If the subclasses are not constrained to be disjoint, their sets of entities may overlap; that is the same entity may be a member of more than one subclass of the specialization.

An example is expected from the student (Example -1 Mark+Description 2 Marks)

(iii) **Covering constraint**

Within an ISA hierarchy, a covering constraint determines where the entities in the subclasses collectively include all entities in the superclass.

An example is expected from the student (Example -1 Mark+Description 2 Marks)

Enhanced ER diagram is not included in the syllabus. Student who has written participation constraints alone with suitable examples can be awarded full marks

7. *Consider the following database with primary keys underlined* *(9 Marks)*

Suppliers (sid, sname, address)
Parts (pid, pname, color)
Catalog (sid, pid, cost)

sid is the key for Suppliers, pid is the key for Parts, and sid and pid together form the key for Catalog. The Catalog relation lists the prices charged for parts by Suppliers.

Write relational algebra for the following queries:-
(i) Find then names of suppliers who supply some red part.
(ii) Find the sids of suppliers who supply some red or green part.
(iii) Find the sids of suppliers who supply some red part and some green part.

(i) $\pi_{sname}(\pi_{sid}((\pi_{pid}\sigma_{color='red'}Parts) \bowtie Catalog) \bowtie Suppliers)$ (3 Marks)

(ii) $\pi_{sid}(\pi_{pid}(\sigma_{color='red' \vee color='green'}Parts) \bowtie catalog)$ (3 Marks)

(iii) $\rho(R1, \pi_{sid}((\pi_{pid}\sigma_{color='red'}Parts) \bowtie Catalog))$ (3 Marks)

$\rho(R2, \pi_{sid}((\pi_{pid}\sigma_{color='green'}Parts) \bowtie Catalog))$

$R1 \cap R2$

Full marks can be awarded to students if they wrote the answer as a single relational algebra expression or using intermediate relations to formulate the final answer.

PART C

(Answer all questions. Each carries 3 marks)

8. *What are the basic data types available for attributes in SQL?* *(3 Marks)*

The basic data types available for attributes are

- Numeric data types – INT, SMALLINT, FLOAT, REAL, DOUBLE PRECISION. (.5 Mark)
- Character string- CHAR(n), VARCHAR(n) (.5 Mark)
- Bit-string – BIT(n), BIT VARYING(n) (.5 Mark)
- Boolean- TRUE, FALSE. (.5 Mark)
- Date and time – YYYY-MM-DD, HH:MM:SS (.5 Mark)
- Timestamp (.5 Mark)

9. *List the aggregate functions in SQL* *(3 Marks)*

(i) Average: **avg** (1 Mark)
(ii) Minimum: **min** (.5 Mark)
(iii) Maximum: **max** (.5 Mark)
(iv) Total: **sum** (.5 Mark)
(v) Count: **count** (.5 Mark)

10. *Let E = {B →A, D→A, AB→D} is a set of Functional Dependencies. Find a minimal cover for E*

The minimal cover for E is {B → D, D→A} (3 Marks)

11. *Define Boyce-Codd normal form(BCNF). Give an example of a relation that is in 3NF but not in BCNF*

A relation schema R is in BCNF if whenever a nontrivial functional dependency X→A holds in R, then X is a superkey of R. (1 Mark)

Following figure shows a relation TEACH that is in 3NF but not BCNF.

TEACH

Student	Course	Instructor
Narayan	Database	Mark
Smith	Database	Navathe
Smith	Operating Systems	Ammar
Smith	Theory	Schulman
Wallace	Database	Mark
Wallace	Operating Systems	Ahamad
Wong	Database	Omiecinski
Zelaya	Database	Navathe
Narayan	Operating Systems	Ammar

The FDs of this relation are:

FD1: {Student, Course} → Instructor

FD2: Instructor → Course.

Because of the presence of FD2 this relation is not in BCNF. (2 Marks)

PART D

(Answer any two questions. Each carries 9 marks)

12. *Consider the following relations for bank database (Primary keys are underlined):*

Customer (customer-name, customer-street, customer-city)

Branch (branch-name, branch-city, assets)

Account (account-number, branch-name, balance)

Depositor (customer-name, account-number)

Loan (loan-number, branch-name, amount)

Answer the following in SQL :

(i) Create tables with primary keys and foreign keys. (5 Marks)

(ii) Create an assertion for the sum of all loan amounts for each branch must be less than the sum of all account balances at the branch. (4 Marks)

(i) **create table** Customer
(customer-name **char**(20),
customer-street **char**(30),
customer-city **char**(30),
primary key (customer-name)); (1 Mark)

create table Branch
(branch-name **char**(15),
branch-city **char**(30),

balance **integer,**

primary key (branch-name)

check (assets>=0)); (1 Mark)

create table Account

(account-number char(10),

branch-name char(15),

balance integer,

primary key (account-number),

foreign key (branch-name) **references** branch,

check (balance >= 0)); (1 Mark)

create table Depositor

(customer-name char(20),

account-number char(10),

primary key (customer-name, account-number),

foreign key (customer-name) **references** Customer,

foreign key (account-number) **references** Account); (1 Mark)

create table Loan

(loan-number **char**(10),

branch-name **char**(15),

amount **interger**,

primary key (loan-number),

foreign key(branch-name) **references** Branch); (1 Mark)

(ii) **create** assertion sum-constraint **check**

(**not exists**(**select** * from Branch

where (**select sum**(amount) **from** Loan

where Loan.branch-name = Branch.branch-name)

>= **(select sum**(balance) **from** Account

where Account.branch-name = Branch.branch-name))); (4 Marks)

13. *Given R(A,B,C,D,E) with the set of FDs, F = {AB→CD, ABC →E, C →A}.*

(i) Find any two candidate keys of R *(3 Marks)*

(ii) What is the normal form of R? Justify your answer. *(6 Marks)*

(i) $(AB)^+$ = ABCDE, CD are included in closure because of the FD AB →CD, and E is included in closure because of the FD ABC →E.

Now $(BC)^+$ = BCAED, A is included in closure because of the FD C →A, and then E is included in closure because of the FD ABC → E and lastly D is included in closure because of the FD AB →CD.

Therefore **two candidate keys are : AB and BC**. (3 Marks)

(ii) The prime attributes are A, B and C and non-prime attributes are D and E. A relation scheme is in 2NF, if all the non-prime attributes are fully functionally dependent on the relation key(s). From the set of FDs we can see that the non- prime attributes (D,E) are fully functionally dependent on the prime attributes, therefore, the relation is in 2NF.

A relation scheme is in 3NF, if for all the non-trivial FDs in F+ of the form X →A, either X is a super key or A is prime. From the set of FDs we see that for all the FDs, this is satisfied, therefore, the relation is in 3NF. A relation scheme is in BCNF, if for all the non-trivial FDs in F+ of the form X→A, X is a super key. From the set of FDs we can see that for the FD C →A, this is not satisfied as LHS is not a super key, therefore, the relation is not in BCNF. *Hence, the given relation scheme is in 3NF.* (6 Marks)

14. *(a) What are Armstrong's axioms*? (3 Marks)

(b) Write an algorithm to compute the attribute closure of a set of attributes (X) under a set of functional dependencies (F). (3 Marks)

(c) Explain three uses of attribute closure algorithm. (3 Marks)

(a) The **Armstrong's axioms** are:

(i) **Reflexivity rule** (1 Mark)

If X is a set of attributes and X ⊇ Y, then X → Y holds.

(ii) **Augmentation rule** (1 Mark)

If X → Y holds and Z is a set of attributes then ZX → ZY holds.

(iii) **Transitivity rule**.

If X → Y holds and Y → Z holds, then X → Z holds. (1 Mark)

(b) **Determining X^+, the Closure of X under F**

Input: A set F of FDs on a relation schema R, and a set of attributes X, which is a subset of R

$X^+ := X$;

repeat

$oldX^+ := X^+$

for each functional dependency Y→Z in F do

if $X^+ \supseteq Y$ then $X^+ := X^+ \cup Z$;

until (X^+ = oldX^+); (3 Marks)

(c) There are several uses of the attribute closure algorithm:

(i) To test if X is a super key, we compute X^+, and check if X^+ contains all attributes of R (1 Mark)

(ii) We can check if a functional dependency X → Y holds by checking if $Y \subseteq X^+$. That is, we compute X^+ by using attribute closure, and then check if it contains Y. (1 Mark)

(iii) It gives us a alternate way to compute F^+ : For each $X \subseteq R$ we find the closure X^+, and for each $S \subseteq X^+$, we output a functional dependency X→S. (1 Mark)

PART E

(Answer any four questions. Each carries 10 marks)

15. *What are the different types of single-level ordered indices? Explain* *(10 Marks)*

Types of Single-level Ordered Indexes:

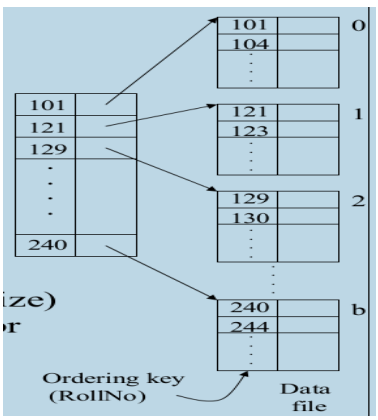
Primary Indexes (1 mark)

Clustering Indexes

Secondary Indexes

(i) **Primary Index**

- Defined on an ordered data file
- The data file is ordered on a **key field**
- Includes one index entry *for each block* in the data file; the index entry has the key field value for the *first record* in the block, which is called the *block anchor*
- A similar scheme can use the *last record* in a block.
- A primary index is a nondense (sparse) index, since it includes an entry for each disk block of the data file (1.5 marks)

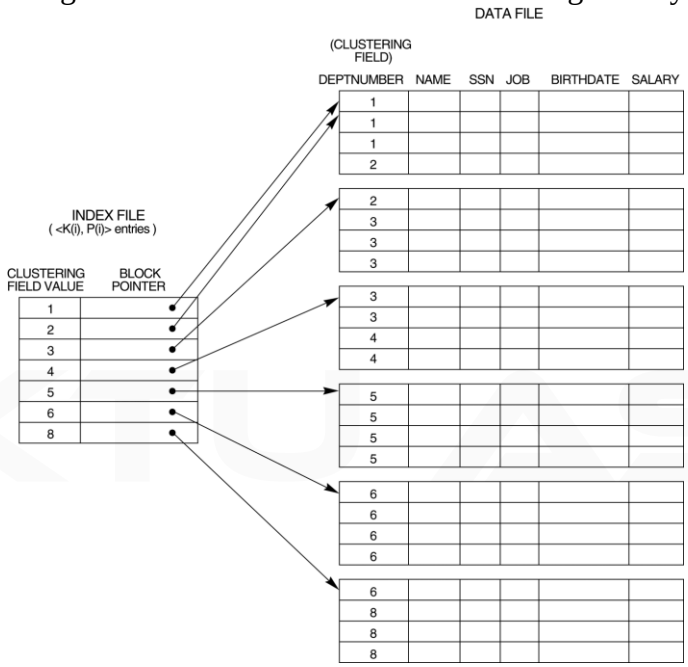


(1.5 marks)

(ii) **Clustering Index**

- Defined on an ordered data file
- The data file is ordered on a *non-key field* unlike primary index, which requires that the ordering field of the data file have a distinct value for each record.
- Includes one index entry *for each distinct value* of the field; the index entry points to the first data block that contains records with that field value.
- It is another example of *nondense* index where Insertion and Deletion is relatively straightforward with a clustering index. (1.5 Marks)

A clustering index on the DEPTNUMBER ordering nonkey field of an EMPLOYEE file



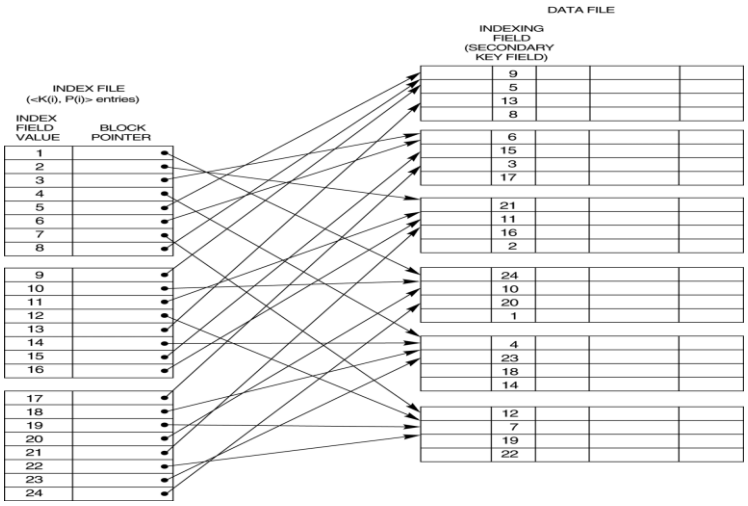
(1.5marks)

(iii) **Secondary Index**

- A secondary index provides a secondary means of accessing a file for which some primary access already exists.
- The secondary index may be on a field which is a candidate key and has a unique value in every record, or a non-key with duplicate values.
- The index is an ordered file with two fields. The first field is of the same data type as some **non-ordering field** of the data file that is an indexing field.
- The second field is either a **block** pointer or a record pointer.
- There can be *many* secondary indexes (and hence, indexing fields) for the same file.
- Includes one entry *for each record* in the data file; hence, it is a dense index

(1.5marks)

A dense secondary index (with block pointers) on a nonordering key field of a file.



(1.5marks)

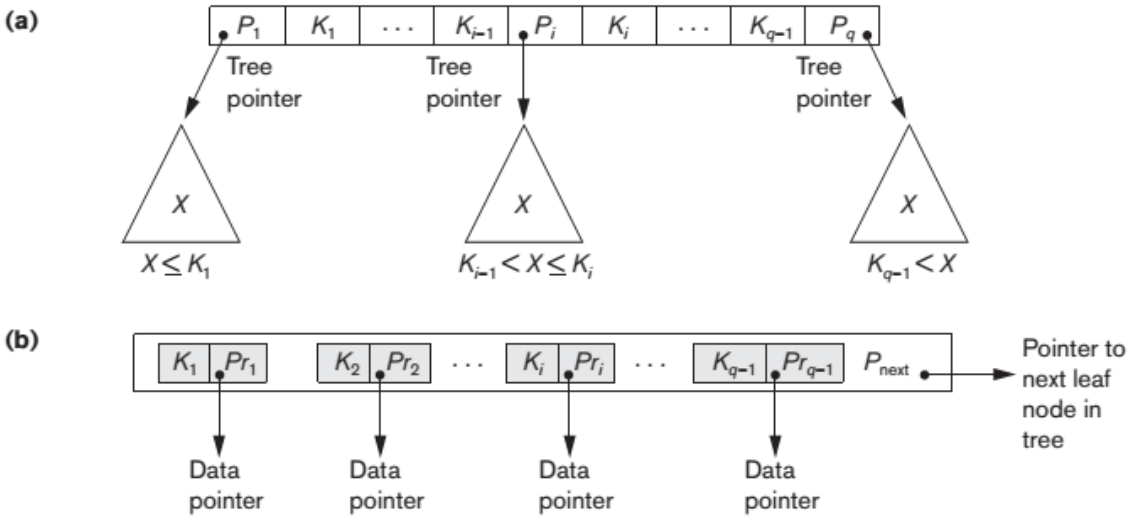
16. *(a) What is a B^+-tree?* *(2 Marks)*

(b)Describe the structure of both internal and leaf nodes of a B^+-tree of order p (8 Marks)

(a) The B+-tree search structure, which is widely used, is a balanced tree in which the internal nodes direct the search and leaf nodes contain the data entries. (2 Marks)

(b) The structure of the internal and leaf nodes of a B+-tree of order p is as follows:

The nodes of a B^+-tree. (a) Internal node of a B^+-tree with $q - 1$ search values. (b) Leaf node of a B^+-tree with $q - 1$ search values and $q - 1$ data pointers.



(2 Marks fig-a, 1 Mark fig-b)

The structure of the internal nodes of a B+- tree of order p (Figure-a) is as follows:

1. $<P_1, K_1, P_2, K_2, \ldots, P_{q-1}, K_{q-1}, P_q>$

   where $q \leq p$ and each P_i is a **tree pointer**.

2. . Within each internal node, $K_1 < K_2 < \ldots < K_{q-1}$.

3. For all search field values X in the subtree pointed at by P_i , we have $K_{i--1} < X \leq K_i$ for $1 < i < q$; $X \leq K_i$ for $i = 1$; and $K_{i-1} < X$ for $i = q$.
4. Each internal node has at most p tree pointers.
5. Each internal node, except the root, has at least Ceil(p/2) tree pointers. The root node has at least two tree pointers if it is an internal node.
6. An internal node with q pointers, $q \leq p$, has $q - 1$ search field values (3 Marks)

The structure of the leaf nodes of a B+-tree of order p (Figure- b) is as follows:

1. Each leaf node is of the form< <K_1 ,P r_1>,<K_2,Pr_2>,...,<K_{q-1},Pr_{q-1}>, P_{next}> where $q \leq p$, each Pr_i is a data pointer, and P_{next} points to the next leaf node of the B^+-tree.
2. Within each leaf node, $K_1 \leq K_2 \ldots , K_{q-1}, q \leq p$.
3. Each Pr_i is a data pointer that points to the record whose search field value is K_i or to a file block containing the record (or to a block of record pointers that point to records whose search field value is K_i if the search field is not a key).
4. . Each leaf node has at least Ceil(p/2)values.
5. All leaf nodes are at the same level. (2 Marks)

17. *Differentiate between static hashing and dynamic hashing*. (10 Marks)

- **Static Hashing** has the number of primary pages in the directory fixed.
- Thus, when a bucket is full, we need an overflow bucket to store any additional records that hash to the full bucket. This can be done with a link to an overflow page, or a linked list of overflow pages.
- The linked list can be separate for each bucket, or the same for all buckets that overflow.
- When searching for a record, the original bucket is accessed first, then the overflow buckets.
- Provided there are many keys that hash to the same bucket, locating a record may require accessing multiple pages on disk, which greatly degrades performance.
- Operation

  (i) Insertion − When a record is required to be entered using static hash, the hash function h computes the bucket address for search key K, where the record will be stored.

  Bucket address = h(K)

  (ii) Search − When a record needs to be retrieved, the same hash function can be used to retrieve the address of the bucket where the data is stored. (5 Marks)

- The problem of lengthy searching of overflow buckets is solved by **Dynamic Hashing**.
- In Dynamic Hashing the size of the directory grows with the number of collisions to accommodate new records and avoid long overflow page chains..

- Extendible and Linear Hashing are two dynamic hashing techniques.

(5 Marks to be awarded if the student clearly explains the above mentioned points)

Hashing is not included in the syllabus. Give appropriate mark if the student attempted the question explaining hashing, hash function, hashing techniques and collision resolution techniques.

18. *How concurrency is controlled using Timestamp Ordering algorithm.* (10 Marks)

Timestamp:

- A monotonically increasing variable (integer) indicating the age of an operation or a transaction. A larger timestamp value indicates a more recent event or operation.
- Timestamp based algorithm uses timestamp to serialize the execution of concurrent transactions. (2 Marks)

Timestamp based concurrency control algorithm

Basic Timestamp Ordering

1. Transaction T issues a write_item(X) operation:
   - If read_TS(X) > TS(T) or if write_TS(X) > TS(T), then an younger transaction has already read the data item so abort and roll-back T and reject the operation.
   - If the condition in part (a) does not exist, then execute write_item(X) of T and set write_TS(X) to TS(T).
2. Transaction T issues a read_item(X) operation:
   - If write_TS(X) > TS(T), then an younger transaction has already written to the data item so abort and roll-back T and reject the operation.
   - If write_TS(X) ≤ TS(T), then execute read_item(X) of T and set read_TS(X) to the larger of TS(T) and the current read_TS(X). (8 Marks)

Time Stamp ordering is not included in syllabus. Can also be awarded marks if the student discussed about need of concurrency control and 2PL.

19. *Explain the concept behind*

(a) Log-Based Recovery. *(5 Marks)*

(b) Deferred Database Modification. *(5 Marks)*

(a) A **log** is kept on stable storage. (1 Mark)

- The log is a sequence of **log records**, and maintains a record of update activities on the database.
- When transaction T_i starts, it registers itself by writing a $<T_i$ **start**>log record
- Before T_i executes **write**(X), a log record $<T_i, X, V_1, V_2>$ is written, where V_1 is the value of X before the write, and V_2 is the value to be written to X.

  Log record notes that T_i has performed a write on data item X_j X_j had value V_1 before the write, and will have value V_2 after the write.
- When T_i finishes it last statement, the log record $<T_i$ **commi**t> is written.
- Two approaches using logs (1) Deferred database modification (2) Immediate database modification (4 Marks)

(b) The **deferred database modification** scheme records all modifications to the log, but defers all the **write**s to after partial commit.

- Assume that transactions execute serially
- Transaction starts by writing $<T_i$ ***start***> record to log.
- A **write**(X) operation results in a log record $<T_i, X, V>$ being written, where V is the new value for X. Note: old value is not needed for this scheme
- The write is not performed on X at this time, but is deferred.
- When T_i partially commits, $<T_i$ **commit**> is written to the log
- Finally, the log records are read and used to actually execute the previously deferred writes.
- Redoing a transaction Ti (**redo**Ti) sets the value of all data items updated by the transaction to the new values. ☒
- Crashes can occur while (i) the transaction is executing the original updates, or (ii) while recovery action is being taken (5 Marks)

20. *(a) What are the components of GIS?* *(3 Marks)*

(b) Explain the characteristics of data in GIS. *(3 Marks)*

(c) What are the constraints in GIS ? *(4 Marks)*

(a) GIS systems can be viewed as an integration of three components:

- Hardware and software (1 Mark)
- Data (1 Mark)
- People (1 Mark)

(b) Characteristics of data in GIS:

- Location
- Temporality

- Complex spatial features
- Thematic values
- Fuzzy objects
- Entity versus field based data
- Generalization
- Roles
- Object ID
- Data quality

(c) Constraints in GIS:

- Topological integrity constraints
- Semantic integrity constraints
- User-defined integrity constraints.
- Temporal constraints.

END