Assignment

1. Explain the informal design guidlines used as measures to determine the quality of relation schema design.

i) Imparting clear Semantics to Attributes in Relations:

The group of attributes belonging to one relation to have certain real-world meaning and a proper interpretation associated with them.

The semantics of a relation refers to its meaning resulating from the interpretation of attribute values in a tuple.

Employee

Ename	SSN	Bdate	Address	Dnumber
	PK			FK

DEPARTMENT

Dname	Dnumber	Dmgr_SSN
	PK	FK

ii). Redundant Information in Tuples and Update Anomalies.

One goal of schema design is to minimize the storage space used by the base relation grouping attribute into relation schema has a significant effect on storage space.

Ex:

EMPLOYEE

EName	SSN	Bdate	Address	Dnumber
Smith	123456	1965-01-9	Houston	5
Wong	333445	1955-12-08	Houston	5
Zelaya	999887	1968-7-19	Spring	4
Wallace	987654	1941-6-20	Bellaire	4
Narayan	666884	1962-9-15	Humble	5

Update anomalies can be classified into insertion anomalies, deletion and modificatio

Insertion anomalies:

To insert a new employee tuple into EMP_DE we must include either the attribute values for the department that the employee work for, or Null.

Deletion anomalies:

If we delete from EMP_DEPT an employee tuple that happen to represent the last employee working for a particular department, the information concerning that department is lost from the database.

Modification anomalies:

In EMP-DEPT, if we change the value of one of the attribute of a particular department say, the manager of department 5 we must update the tuple of all employee who work

in that department; otherwise, the database will become inconsistent.

iii). Null values in tuple:

If many of the attributes do not apply to all tuples in the relation, we end up with many NULL's in those tuples. This can waste space at the storage level & may also lead to problems with understand the meaning of the attribute.

Select and Join operation involve comparison if Null values are present, the result may become unpredictable.

iv). Generation of Spurious tuples:

Consider the relation schemas EMP_LOCS and EMP_PROJ1. Suppose if we perform Natural Join operation on EMP_PROJ1 and EMP_LOCS the result produces many more tuples than the original set of tuples.

These additional tuples are called spurious tuples because they represent spurious information that is not valid.

2. Define functional dependency. Explain with an example.

A. A functional dependencies, denoted by $X \rightarrow Y$ between two sets of attributes X and Y that are subset of R specifies a constraint on the tuples in a relation state r of R. The constraint is that, for any two tuples t1 and t2 in r that have $t1[X] = t2[X]$, they must also have $t1[Y] = t2[Y]$

Ex: Consider the relation schema EMP-PROJ from the semantics of the attributes and the relation:

SSN → ENAME
Pnumber → {Pname, Plocation}
{SSN, Pnumber} → Hours.

a) the value of an employee's social security number uniquely determines the Ename.

b) the value of a Pnumber uniquely determine the Pname and Plocation.

c) Combination of SSN and Pnumber values uniquely determines the number of hours the employee currently works on the project per week [Hours].

3. Define Normal form. Explain 1NF, 2NF and 3NF with suitable ~~one~~ example for each.

A. The normal form of a relation refers to the highest normal form condition that it meets and hence indicates the degree to which it has been normalized.

1 Normal form:

States that the domain of an attribute must include only atomic values and that the value of any attribute in a tuplee must be a single value from the domain of that attribute.

Ex: Relation EMPLOYEE is not in 1NF because of multi-valued attribute EMP_PHONE

EMP_ID	EMP_Name	EMP_PHONE	EMP_STATE
14	John	7272826385	UP
20	Harry	8745783832	Bihar
12	Sam	8589830302	Punjab

Second Normal form

Relational must be in 1NF

In the second normal form, all non-key attributes are fully functional dependent on the primary key.

Ex: Let's assume, a school can store the data of teachers & the subjects they teach. In a school, a teacher can teach more than one subject.

Teacher table

Teacher_ID	Subject	Teacher_Age
25	Chemistry	30
25	Biology	30
47	Math	35
83	English	38
83	Computer	38

Teacher_Detail table

Teacher_ID	Teacher_Age
25	30
47	35
83	38

Teacher_Subject table

Teacher_ID	Subject
25	Chemistry
25	Biology
47	Math
83	English
83	Computer

Third Normal form:

A relation will be in 3NF if it is in 2NF form and not contain any transitive partial dependency.

3NF is used to reduce the data duplication

ex;

EMPLOYEE table

EMP_ID	EMP_Name	EMP_ZIP
222	Harvey	201010
333	Stephen	02228
444	Ian	60007
555	Katharine	06389
666	John	42007

EMPLOOYE ZIP table

EMP_ZIP	EMP_STATE	EMP_CITY
201010	UP	Noida
02228	US	Boston
60007	US	Chicago
06389	UK	Norwich
42007	MP	Bhopal

4. Define multi valued Dependency. Explain 4NF and 5NF with examples:

1. Multivalued dependency occurs when two attributes in a table are independent of each other but, both depend on a third attribute.

classmate
Date
Page

Fourth normal form (4NF)

A relation will be in 4NF if it is in Boyce codd normal form and has no multi-valued dependency.

For a dependency A → B, if for a single value of A, multiple values of B exists then the relation will be a multi-valued dependency.

Ex:

STUDENT_Course

STU_ID	Course
21	Computer
21	Math
34	Chemistry
74	Biology
59	Physics

STUDENT_HOBBY

STU_ID	HOBBY
21	Dancing
21	Singing
34	Dancing
74	Cricket
59	Hockey

Fifth normal form:

A relation is in 5NF if it is in 4NF and not contains any join dependency and joining should be lossless.

5NF is also known as Project-join normal form.

Ex:

Semester	Subject
S1	Computer
S2	Math
S1	Chemistry
S2	Math

P2

Subject	Lecturer
Computer	Anshika
Computer	John
Math	John
Math	Akash
Chemistry	Praveen

P3

Semester	Lecturer
S1	Anshika
S1	John
S1	John
S2	Akash

5. Define Minimal cover. Write an algorithm for finding a minimal cover F for a set of functional dependencies E. Find the minimal cover for the given set of FD's be. E: {B→A, D→A, AB→D}.

A. A minimal cover of a set of functional dependencies E is a set of functional dependa

F that satisfies the property that every dependency in E is in the closure F^+ of F.

Algorithm:

1. Set $F = E$
2. Replace each functional dependency $X \to \{A_1, A_2, \ldots, A_n\}$ in F by the n functional dependencies $X \to A_1, X \to A_2, \ldots, X \to A_n$.

3. For each functional dependency $X \to A$ in F for each attribute B that is an element of X if $\{\{F - \{X \to A\}\} \cup \{(X - \{B\}) \to A\}\}$ is equivalent to F then replace $X \to A$ with $(X - \{B\}) \to A$ in F.

4. For each remaining functional dependency $X \to A$ in F if $\{F - \{X \to A\}\}$ is equivalent to F, then remove $X \to A$ from F.

E: $\{B \to A, D \to A, AB \to D\}$

Step 1: Break down the RHS of each functional dependency into a single attribute, here all RHS attributes are single.

Step 2: We need to determine if $AB \to D$ has any redundant attribute on the left hand side; that is, can it be replaced by $B \to D$ or $A \to D$?
Consider $B \to A$, by augmenting B on both sides we have $BB \to AB$ or $B \to AB$.
Hence by the transitive rule we get $B \to D$

Because $B \to AB$ and $AB \to D$.

We now have $E = \{B \to A, D \to A, B \to D\}$. No further reduction is possible in step 2 since all FD's have a single attribute on the left hand side.

Step 3: we look for a redundant FD in E, By using the transitive rule on $B \to D$ & $D \to A$. we derive $B \to A$, Hence $B \to A$ is redundant in E & can be diminated.

Therefore, the minimal cover of E is $\{B \to D, D \to A\}$.

7. What are the ~~interfaces~~ interference rules on FD's ? How they are useful ? Explain with an example.

A. The set of functional dependencies are specified by F on relation schema R, other functional dependencies can be inferred or deduced from the FD's in F.

For example: Department has one manager, the Dept_no uniquely determines Mgr_SSN, and manager uniquely determines ph no called Mgr_phone then these two dependies together imply that Dept_no → Mgr_phone.
(Dept_no → Mgr_SSN)
(Mgr_SSN → Mgr_Phone)
Dept_No → Mgr_phone

Therefore, it is useful to define a concept called closure formally that includes all possible dependencies that can be inferred from the given set F.

The closure F+ of F is the set of function dependencies that can be inferred from F. To determine a systematic way to infer dependencies, the set of inference rules are used to infer new dependencies from a given set of dependencies.

8. Why Concurrency control is needed demonstrate with an example.

1. Several problems can occur when concurrent transactions execute in an uncontrolled manner. We illustrate some of these problems by referring to a much simplified airline reservations database in which a record is stored for each airline flight.

Each record includes the number of reserved seats on that flight as a named data item, among other information. T1 that can transfers N reservation from one flight whose number of reserved seats is stored in the database item named X to another flight whose number of reserved seats is stored in the database item named X.

T_1	T_2

read_item(X);
X = X-N;
write_item(X);
read_item(Y);
Y = Y+N;
write_item(Y);
	read_item(X);
	X = X+M;
	write_item(X);

9. Discuss the Desirable properties of transactions

1. Atomicity : A transaction is an atomic unit of processing. It should either be performed in its entirety or not performed at all.

Consistency preservation : A transaction should be consistency preserving, meaning that if it is completely executed from begening to end without interference from other transaction it should take the database from one consistent state to another.

Isolation : A transaction should appear as though it is being executed in isolation from other transactions, even though many transactions are executing concurrently. That is the execution of a transaction should not be interfered with by any other transactions executing concurrently.

Durability or permanency : The changes applied to the database by a committed transaction must persist in the database

These changes must not be lost because of any failure.

10. What is schedule & Explain conflict serializable schedule with example.

A. When transactions are executing concurrent in an interleaved fashion, then the order of execution of operation from all the various transactions is known as a schedule.

Two schedules are said to be conflict equivalent if the order of any two conflicting operations is the same in both schedules. Two operations in a schedule are said to conflict if they belong to different transaction, access the same database item, and either both are write_item operation or one is a write_item and the other a read_item.

Using the notion of conflict equivalence we define a schedule S to be conflict serializable if it is equivalent to some serial schedule S'. In such a case, we can reorder the non conflicting operation in S until we form the equivalent serial schedule.

Ex:

T_1	T_2
read_item (X); X = X - N;	
	read_item (X) X = X + m;
write_item (X); read_item (Y);	
	write_item (X);
Y = Y + N; write_item (Y);	

T_1	T_2
read_item (X); X = X - N; write_item (X);	
	read_item (X); X = X + M; write_item (X);
read_item (Y); Y = Y + N; write_item (Y);	

11. Briefly discuss the two phase locking protocol used in concurrency control.

A. Types of lock and System lock tables.

Types of locks used in concurrency control are binary locks, shared/exclusive locks - also known as read/write locks and a certify lock that improves performance of locking protocols.

Binary locks:

- A binary lock can have two states: locked and unlocked.

- A distinct lock is associated with each database item x. If the value of the lock on x is 1, item x cannot be accessed by a database operation that request the item. If the value of the lock on x is 0, the item can be accessed when requested & the lock value is changed to 1.

The current value of the lock associated with item x is referred to as lock(x).

Two operation, lock_item and unlock_item are used with binary locking.

```
lock_item(X):
B: if lock(X)=0
   then lock(X)←1
else
   Begin
   wait (until lock(X)=0
      and the lock manages wakes up the
      transaction);
   go to B
   end;
unlock_item(X):
   lock(X)←0;
   if any transaction are waiting
   then wakeup one of the waiting transactions;
```

→. Shared / Exclusive locks: The preceding binary locking scheme is too restrictive for database items because at most, one transaction can hold a lock on a given item. We should allow several transactions to access the same item X if they all access X for reading purposes only. This is because read operations on the same item by different transactions are not conflicting. However, if a transaction is to write an item X, it must have exclusive access to X.

The system must enforce the following rules:

A transaction T must issue the operation read_lock(X) or write_lock(X) before any read_item(X) operation is performed in T.

A transaction T must issue the operation unlock(X) after all read_item(X) and write_item(X) are completed in T.

12. Explain how shadow paging helps to recover from transaction failure.

A. This recovery scheme does not require the use of a log in a single-user environment. In a multiuser environment, a log may be needed for the consistency control method.

Shadow paging considers the database to be

made up of a number of fixed size disk page say n for recovery purpose.

A directory with n entries is constructed where the ith entry points to the ith database page on disk.

The directory is kept in main memory if it is not too large, and all references - read or write to database pages on disk go through it.

When a transaction begins, executing, the current directory whose entries point to the most recent or current database pages on disk is copied into a shadow directory.

The shadow directory is then saved on disk while the current directory is used by the transaction.