SQL Fundamentals

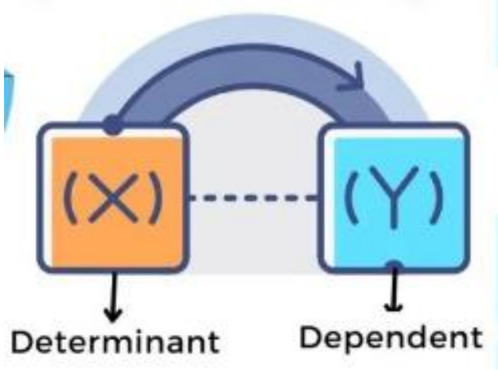
COURSE : 23CAT- Database Management System
UNIT I : Introduction
CLASS : I Semester / I MCA

Evils of Redundancy

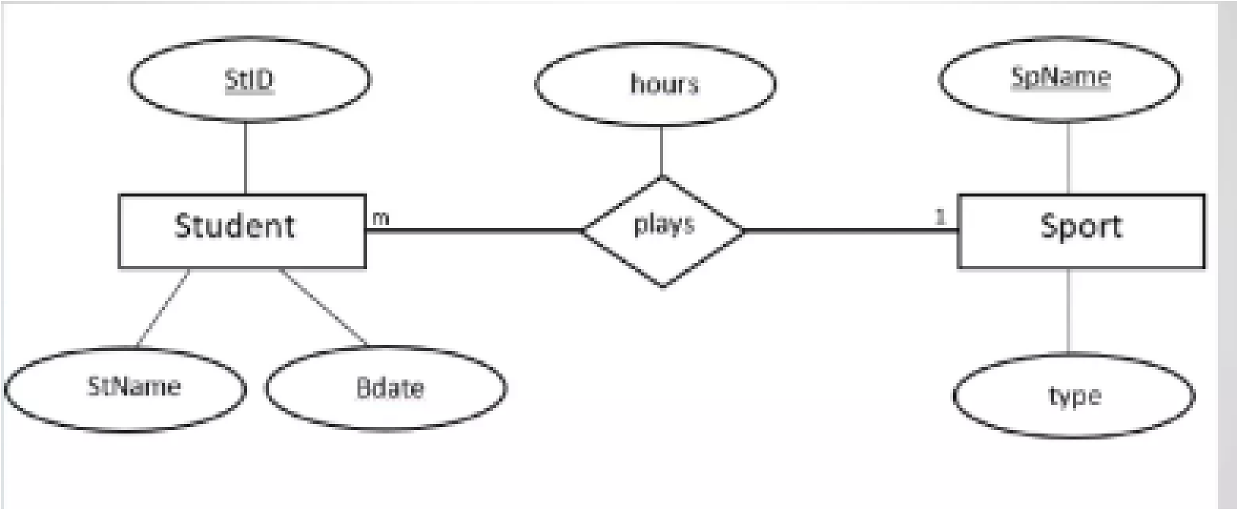
- Redundancy is at the root of several problems associated with relational schemas:
  - redundant storage, insert/delete/update anomalies
- Integrity constraints, in particular functional dependencies, can be used to identify schemas with such problems and to suggest refinements
- Main refinement technique: **decomposition** (replacing ABCD with, say, AB and BCD, or ACD and ABD)
- Decomposition should be used judiciously:
  - Is there reason to decompose a relation?
  - What problems (if any) does the decomposition cause?

Replacing a relation with a collection of `smaller' relations

- Functional dependency in DBMS refers to the relationship between attributes in a database table
- The functional dependency is a relationship that exists between two attributes
- It typically exists between the primary key and non-key attribute within a table
  - X → Y
- For example,
  - Emp_Id → Emp_Name



- A Functional dependency is a constraint that describes relationship between attributes in the relation
- FDS and keys are used to define the normal form for relation
- Example:
  - If any two relations agree on the attributes
  - $A_1, A_2, A_3 \ldots A_n$
  - Then, they must also agree on the attributes
  - $B_1, B_2, B_3 \ldots B_n$
  - Formally:
    - $A_1, A_2, A_3 \ldots A_n \longrightarrow B_1, B_2, B_3 \ldots B_n$


StID
hours
SpName
Student
m
plays
1
Sport
StName
Bdate
type

Functional Dependencies(FD)

Type of FD

- **Trivial FD**
  - A trivial functional dependency occurs when Q is a subset of P
- **Non-trivial FD**
  - A non-trivial functional dependency occurs when Q is not a subset of P
- **Completely Non-trivial FD**
  - functional dependency occurs when P intersection Q is null

Armstrong's Axioms property

- Armstrong's axioms are used to conclude functional dependencies on a relational database.
- The inference rule is a type of assertion. It can apply to a set of FD to derive other FD.
- Using the inference rule, we can derive additional functional dependency from the initial set

- Augmentation
  - PR -> QR, if P -> Q
- Reflexivity
  - P -> Q, if Q is a subset of P
- Transitivity
  - If P -> Q and Q -> R, then P -> R i.e. a transitive relation
- Union
  - If X → Y and X → Z then X → YZ
- Decomposition
  - If X → YZ then X → Y and X → Z

- Consider relation obtained from Hourly_Emps:
  - Hourly_Emps (ssn, name, lot, rating, hrly_wages, hrs_worked)
  - Wages(rating, hourly wages)
- Notation: We will denote this relation schema by listing the attributes: SNLRWH
  - This is really the set of attributes {S,N,L,R,W,H}.
  - Sometimes, we will refer to all attributes of a relation by using the relation name. (e.g., Hourly_Emps for SNLRWH)
- Some FDs on Hourly_Emps:
  - ssn is the key: S→ SNLRWH
  - rating determines hrly_wages: R→ W

Example

Wages

R	W
8	10
5	7

- Problems due to R → W :
  - Update anomaly: Can we change W in just
  - the 1st tuple of SNLRWH?
  - Insertion anomaly: What if we want to insert an
  - employee and don’t know the hourly wage for his rating?
  - Deletion anomaly: If we delete all employees with rating 5, we lose the information about the
  - wage for rating 5!

Wages

R	W
8	10
5	7

Hourly_Emps2

S	N	L	R	H
123-22-3666	Attishoo	48	8	40
231-31-5368	Smiley	22	8	30
131-24-3650	Smethurst	35	5	30
434-26-3751	Guldu	35	5	32
612-67-4134	Madayan	35	8	40

S	N	L	R	W	H
123-22-3666	Attishoo	48	8	10	40
231-31-5368	Smiley	22	8	10	30
131-24-3650	Smethurst	35	5	7	30
434-26-3751	Guldu	35	5	7	32
612-67-4134	Madayan	35	8	10	40

Reasoning About FDs

- Given some FDs, we can usually infer additional FDs:
  - ssn $\rightarrow$ did, did $\rightarrow$ lot implies ssn $\rightarrow$ lot
- An FD *f* is *implied by* a set of FDs F if *f* holds whenever all FDs in F hold.
  - F+ = closure of F is the set of all FDs that are implied by F.
- Armstrong's Axioms (X, Y, Z are sets of attributes):
  - *Reflexivity:* If $X \subseteq Y$, then $Y \rightarrow X$
  - *Augmentation*: If $X \rightarrow Y$, then $XZ \rightarrow YZ$ for any Z
  - *Transitivity*: If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$
- These are *sound* and complete inference rules for FDs!

- Couple of additional rules (that follow from AA):
  - Union: If $X \to Y$ and $X \to Z$, then $X \to YZ$
  - Decomposition: If $X \to YZ$, then $X \to Y$ and $X \to Z$
- Example: Contracts(cid,sid,jid,did,pid,qty,value), and:
  - C is the key: $C \to CSJDPQV$
  - Project purchases each part using single contract: $JP \to C$
  - Dept purchases at most one part from a supplier: $SD \to$ P
- $JP \to C$, $C \to CSJDPQV$ imply $JP \to CSJDPQV$
- $SD \to P$ implies $SDJ \to JP$
- $SDJ \to JP$, $JP \to CSJDPQV$ imply $SDJ \to CSJDPQV$

- Computing the closure of a set of FDs can be expensive. (Size of closure is exponential in # attrs!)
- Typically, we just want to check if a given FD $X \rightarrow Y$ is in the closure of a set of FDs F. An efficient check:
- Compute attribute closure of X (denoted X^+) wrt F:
- Set of all attributes A such that $X \rightarrow A$ is in F^+
- There is a linear time algorithm to compute this.
- Check if Y is in X^+
- Does $F = \{A \rightarrow B,\ B \rightarrow C,\ C\,D \rightarrow E\}$ imply $A \rightarrow E$?
- i.e, is $A \rightarrow E$ in the closure F^+ ? Equivalently, is E in A^+ ?

- Schema refinement, Whether any refinement is needed?
- If a relation is in a certain normal form (BCNF, 3NF etc.), it is known that certain kinds of problems are avoided/minimized. This can be used to help us decide whether decomposing the relation will help.
- Role of FDs in detecting redundancy:
  - Consider a relation R with 3 attributes, ABC.
    - No FDs hold: There is no redundancy here.
    - Given $A \rightarrow B$: Several tuples could have the same A value, and if so, they'll all have the same B value!