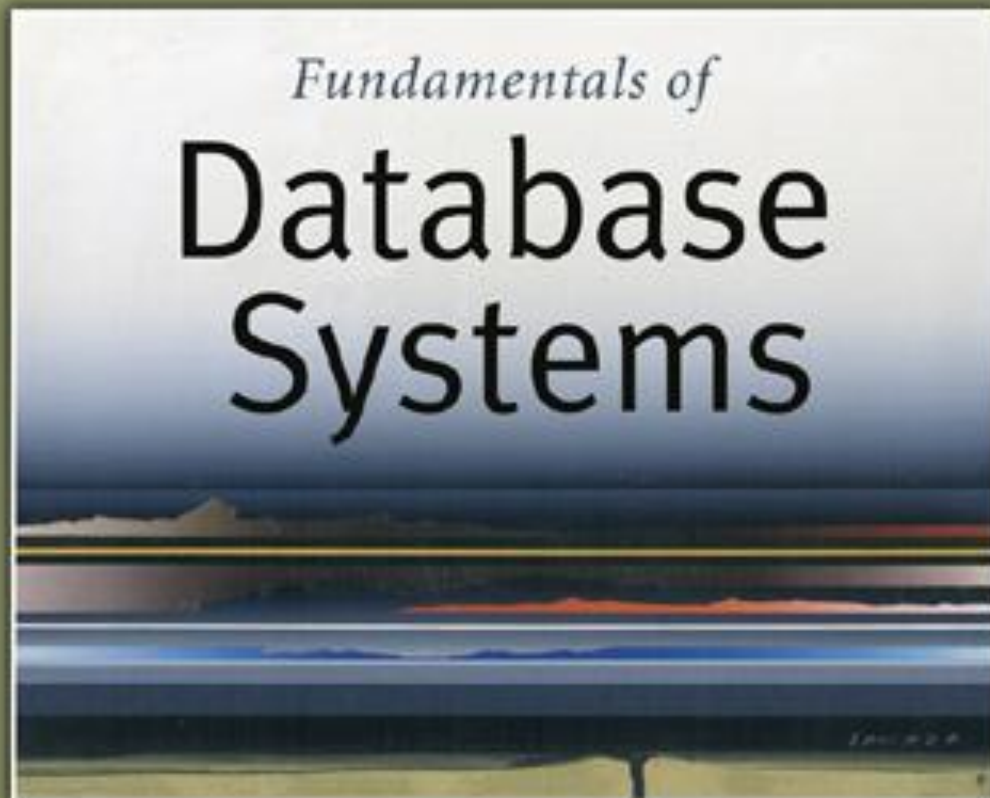
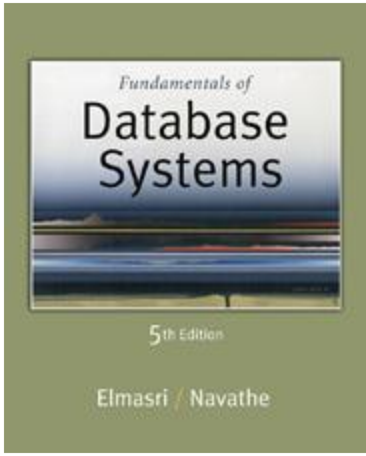


7th Edition

Elmasri / Navathe

Lecture 9

Functional Dependencies and Normalization for Relational Databases



There are two ways to design database

1. **Top-Down method**: By using Entity-Relationship diagram.

2. **Bottom-Up method**: By using Function dependency between attributes and normalization.

- **Fully normalized** term will be used to describe a collection of tables that are structured so that they can not contain redundant data

Informal Design Guidelines for Relational Databases

- What is relational database design?
  - The grouping of attributes to form "good" relation schemas
- Two levels of relation schemas
  - The logical "user view" level
  - The storage "base relation" level
- Design is concerned mainly with base relations
- What are the criteria for "good" base relations?

1. Semantics of the Relation Attributes

- GUIDELINE 1: Informally, each tuple in a relation should represent one entity or relationship instance. (Applies to individual relations and their attributes).
  - Attributes of different entities (EMPLOYEEs, DEPARTMENTs, PROJECTs) should not be mixed in the same relation
  - Only foreign keys should be used to refer to other entities
  - Entity and relationship attributes should be kept apart as much as possible.
- Bottom Line: *Design a schema that can be explained easily relation by relation. The semantics of attributes should be easy to interpret.*

A simplified COMPANY relational database schema

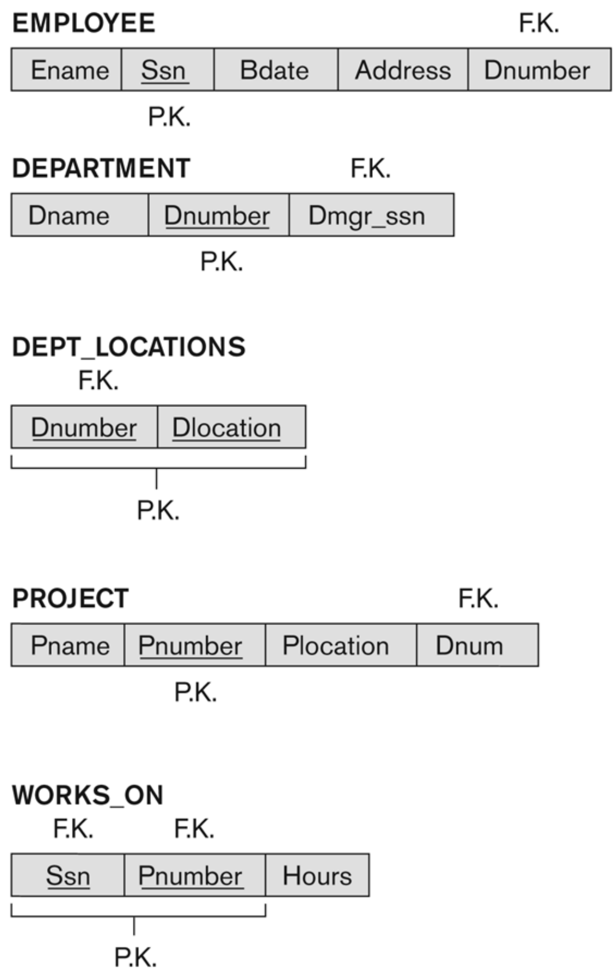


Figure 10.1
A simplified COMPANY relational database schema.

Figure 10.2
Example database state for the relational database schema of Figure 10.1.

EMPLOYEE

Ename	Ssn	Bdate	Address	Dnumber
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX	5
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4
Wallace, Jennifer S.	987654321	1941-06-20	291Berry, Bellaire, TX	4
Narayan, Ramesh K.	666884444	1962-09-15	975 Fire Oak, Humble, TX	5
English, Joyce A.	453453453	1972-07-31	5631 Rice, Houston, TX	5
Jabbar, Ahmad V.	987987987	1969-03-29	980 Dallas, Houston, TX	4
Borg, James E.	888665555	1937-11-10	450 Stone, Houston, TX	1

DEPARTMENT

Dname	Dnumber	Dmgr_ssn
Research	5	333445555
Administration	4	987654321
Headquarters	1	888665555

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Ssn	Pnumber	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	Null

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

2. Redundant Information in Tuples and Update Anomalies

- Information is stored redundantly
  - Wastes storage
  - Causes problems with update anomalies
    - Insertion anomalies
    - Deletion anomalies
    - Modification anomalies

EXAMPLE OF AN UPDATE ANOMALY

- Consider the relation:
  - EMP_PROJ (<u>Emp#, Proj#,</u> Ename, Pname, No_hours)
- Update Anomaly:
  - Changing the name of project number P1 from “Billing” to “Customer-Accounting” may cause this update to be made for all 100 employees working on project P1.

EXAMPLE OF AN INSERT ANOMALY

- Consider the relation:
  - EMP_PROJ(Emp#, Proj#, Ename, Pname, No_hours)
- Insert Anomaly:
  - Cannot insert a project unless an employee is assigned to it.
- Conversely
  - Cannot insert an employee unless a he/she is assigned to a project.

EXAMPLE OF AN DELETE ANOMALY

- Consider the relation:
  - EMP_PROJ(Emp#, Proj#, Ename, Pname, No_hours)
- Delete Anomaly:
  - When a project is deleted, it will result in deleting all the employees who work on that project.
  - Alternately, if an employee is the sole employee on a project, deleting that employee would result in deleting the corresponding project.

Guideline to Redundant Information in Tuples and Update Anomalies

- GUIDELINE 2:
  - Design a schema that does not suffer from the insertion, deletion and update anomalies.
  - If there are any anomalies present, then note them so that applications can be made to take them into account.

3. Null Values in Tuples

- GUIDELINE 3:
  - Relations should be designed such that their tuples will have as few NULL values as possible
  - Attributes that are NULL frequently could be placed in separate relations (with the primary key)
- Reasons for nulls:
  - Attribute not applicable or invalid
  - Attribute value unknown (may exist)
  - Value known to exist, but unavailable

Functional Dependencies (1)

- Functional dependencies (FDs)
  - Are used to specify *formal measures* of the "goodness" of relational designs
  - And keys are used to define **normal forms** for relations
  - Are **constraints** that are derived from the *meaning* and *interrelationships* of the data attributes
- A set of attributes X *functionally determines* a set of attributes Y if the value of X determines a unique value for Y

Functional Dependencies (2)

- $X \rightarrow Y$ holds if whenever two tuples have the same value for X, they *must have* the same value for Y
  - For any two tuples t1 and t2 in any relation instance r(R): If t1[X]=t2[X], *then* t1[Y]=t2[Y]
- $X \rightarrow Y$ in R specifies a *constraint* on all relation instances r(R)
- Written as $X \rightarrow Y$; can be displayed graphically on a relation schema as in Figure (denoted by the arrow).
- FDs are derived from the real-world constraints on the attributes

Two relation schemas suffering from update anomalies

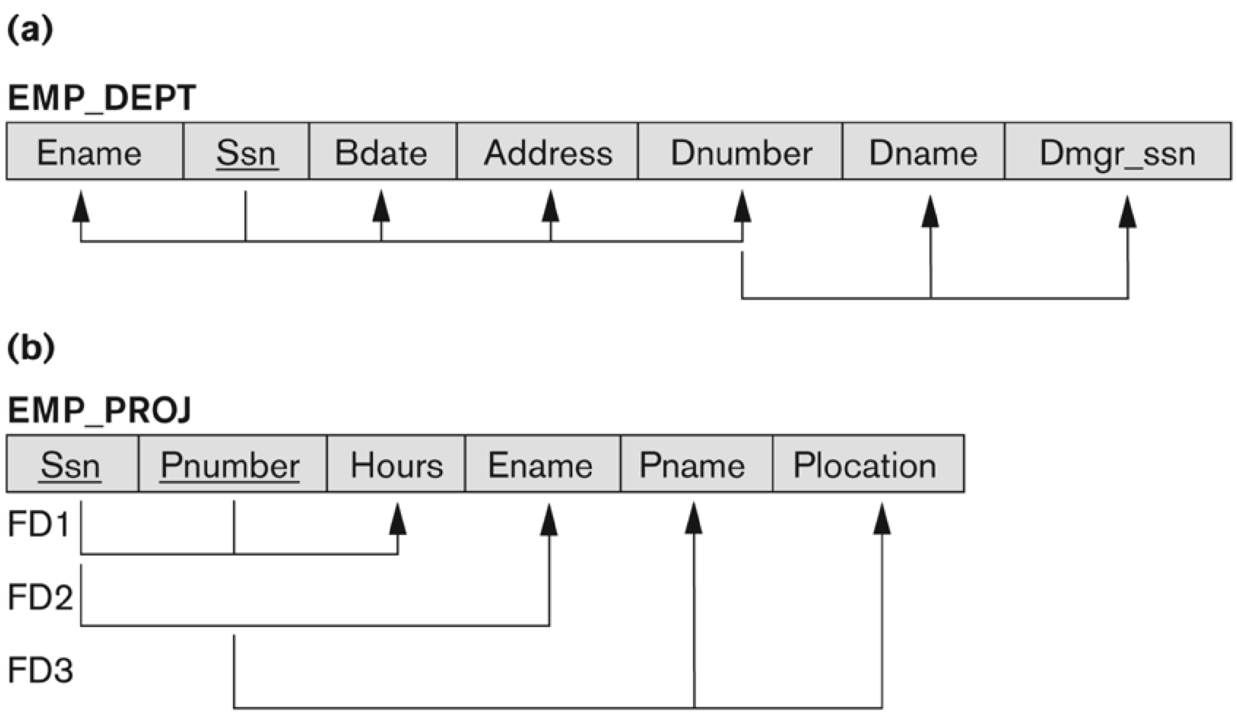


Figure 10.3
Two relation schemas suffering from update anomalies.
(a) EMP_DEPT and
(b) EMP_PROJ.

Examples of FD constraints (1)

- Social security number determines employee name
  - SSN → ENAME
- Project number determines project name and location
  - PNUMBER → {PNAME, PLOCATION}
- Employee ssn and project number determines the hours per week that the employee works on the project
  - {SSN, PNUMBER} → HOURS

Examples of FD constraints (2)

- An FD is a property of the attributes in the schema R
- The constraint must hold on *every* relation instance r(R)
- If K is a key of R, then K functionally determines all attributes in R
  - (since we never have two distinct tuples with t1[K]=t2[K])

FD's are a property of the meaning of data and hold at all times: certain FD's can be ruled out based on a given state of the database

TEACH

Teacher	Course	Text
Smith	Data Structures	Bartram
Smith	Data Management	Martin
Hall	Compilers	Hoffman
Brown	Data Structures	Horowitz

Figure 10.7
A relation state of TEACH with a *possible* functional dependency TEXT → COURSE. However, TEACHER → COURSE is ruled out.

Inference Rules for FDs (1)

- Given a set of FDs F, we can **infer** additional FDs that hold whenever the FDs in F hold
- Armstrong's inference rules:
  - IR1. (**Reflexive**) If Y *subset-of* X, then $X \rightarrow Y$
  - IR2. (**Augmentation**) If $X \rightarrow Y$, then $XZ \rightarrow YZ$
    - (Notation: XZ stands for X U Z)
  - IR3. (**Transitive**) If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$

- IR1, IR2, IR3 form a **sound** and **complete** set of inference rules
  - These are rules hold and all other rules that hold can be deduced from these

Inference Rules for FDs (2)

- Some additional inference rules that are useful:
  - **IR4: Decomposition:** If $X \rightarrow YZ$, then $X \rightarrow Y$ and $X \rightarrow Z$
  - **IR5: Union:** If $X \rightarrow Y$ and $X \rightarrow Z$, then $X \rightarrow YZ$
  - **IR6: Psuedotransitivity:** If $X \rightarrow Y$ and $WY \rightarrow Z$, then $WX \rightarrow Z$

- The last three inference rules, as well as any other inference rules, can be deduced from IR1, IR2, and IR3 (completeness property)

EXAMPLE

For a relation R with attributes A, B, C, D, E, F, and the FDs are:

$$A \rightarrow BC$$

$$B \rightarrow E$$

$$CD \rightarrow EF$$

Show that $AD \rightarrow F$ holds also for R.

SOLUTION

1.	$A \rightarrow BC$	(given)
2.	$A \rightarrow C$	(decomposition of 1)
3.	$AD \rightarrow CD$	(augmentation of 2)
4.	$CD \rightarrow EF$	(given)
5.	$AD \rightarrow EF$	(transitivity 3,4)
6.	$AD \rightarrow F$	(decomposition of 5)

Inference Rules for FDs (3)

- **Closure** of a set F of FDs is the set F^+ of all FDs that can be inferred from F
- **Closure** of a set of attributes X with respect to F is the set X^+ of all attributes that are functionally determined by X
- X^+ can be calculated by repeatedly applying IR1, IR2, IR3 using the FDs in F

Example

- If:

$SSN \rightarrow ENAME$

$PNUMBER \rightarrow PNAME, PLOCATION$

$\{SSN, PNUMBER\} \rightarrow HOURS$

- Then:

$\{SSN\}^+ = \{SSN, ENAME\}$

$\{PNUMBER\}^+ = \{PNUMBER, PNAME, PLOCATION\}$

$\{SSN, PNUMBER\}^+ = \{SSN, ENAME, PNUMBER, PNAME, PLOCATION, HOURS\}$

Equivalence of Sets of FDs

- Two sets of FDs F and G are **equivalent** if:
  - Every FD in F can be inferred from G, and
  - Every FD in G can be inferred from F
  - Hence, F and G are equivalent if $F^+ = G^+$
- Definition (**Covers**):
  - F **covers** G if every FD in G can be inferred from F
    - (i.e., if G^+ *subset-of* F^+)
- F and G are equivalent if F covers G and G covers F
- There is an algorithm for checking equivalence of sets of FDs

Minimal Sets of FDs (1)

- A set of FDs is **minimal** if it satisfies the following conditions:
  1. Every dependency in F has a single attribute for its RHS.
  2. We cannot remove any dependency from F and have a set of dependencies that is equivalent to F.
  3. We cannot replace any dependency $X \rightarrow A$ in F with a dependency $Y \rightarrow A$, where Y proper-subset-of X (Y subset-of X) and still have a set of dependencies that is equivalent to F.

Minimal Sets of FDs (2)

- Every set of FDs has an equivalent minimal set
- There can be several equivalent minimal sets
- There is no simple algorithm for computing a minimal set of FDs that is equivalent to a set F of FDs
- To synthesize a set of relations, we assume that we start with a set of dependencies that is a minimal set

Computing the Minimal Sets of FDs

Let the given set of FDs be E :

$\{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$.

We have to find the minimum cover of E.

1- All above dependencies are in canonical form (that is RHS is one attribute) ; so we have completed step 1

2- In step 2 we need to determine if $AB \rightarrow D$ has any redundant attribute on the left-hand side; that is, can it be replaced by $B \rightarrow D$ or $A \rightarrow D$?

- Since B $\rightarrow$ A, (augmenting with B on both sides (IR2)), we have $BB \rightarrow AB$, or $B \rightarrow AB$ (i).
- However, $AB \rightarrow D$ as given (ii).
- Hence by the transitive rule (IR3), we get from (i) and (ii), $B \rightarrow D$.
- Hence $AB \rightarrow D$ may be replaced by $B \rightarrow D$.

We now have a set equivalent to original E, say E' : $\{B \rightarrow A, D \rightarrow A, B \rightarrow D\}$.

No further reduction is possible in step 2 since all FDs have a single attribute on the left-hand side.

- In step 3 we look for a redundant FD in E′. By using the transitive rule on

$B \rightarrow D$ and $D \rightarrow A$, we derive $B \rightarrow A$. Hence $B \rightarrow A$ is redundant in E' and can

be eliminated.

- Hence the minimum cover of E is

$\{B \rightarrow D, D \rightarrow A\}$ ➔ table1(<u>B</u>,D), table2(<u>D</u>,A)

Another example

For a relation R with attributes A, B, C, D and the FDs are:

1. **$A \rightarrow BC$**
2. **$B \rightarrow C$**
3. **$A \rightarrow B$**
4. **$AB \rightarrow C$**
5. **$AC \rightarrow D$**

Compute a minimal set of FDs.

Solution

- Apply rule 1 to FD no.1 :

  1. $A \rightarrow B$
  2. $A \rightarrow C$
  3. $B \rightarrow C$
  4. $A \rightarrow B$
  5. $AB \rightarrow C$
  6. $AC \rightarrow D$

FD 1 and 4 are the same, eliminate one.

- FD 6 : We can eliminate C from LHS as it is redundant for: $A \rightarrow C$, then $A \rightarrow AC$ (augmentation);

  $AC \rightarrow D$ (FD 6) Then $A \rightarrow D$ (transitivity)

- FD 5 : We can eliminate it as :

  $A \rightarrow C$, then $AB \rightarrow CB$ (augmentation);

  so $AB \rightarrow C$ (decomposition)
- From 1,3 there exist transitivity that lead to

  $A \rightarrow C$, which is the same as no.2, so eliminate FD 2.

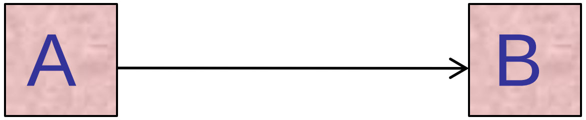
- Then the final minimal set of FDs is:

$\mathbf{A \rightarrow B}$	*Table1 (<u>A</u>, B)*	→	*Table1(<u>A</u>,B,D)*
$\mathbf{B \rightarrow C}$	*Table2 (<u>B</u>, C)*		*Table2 (<u>B</u>, C)*
$\mathbf{A \rightarrow D}$	*Table3 (<u>A</u>, D)*		

Determinancy Diagrams

- A simple way of showing determinants and the attributes that determine is to draw ***determinancy diagram*** (or ***functional dependency diagram***) as follow:

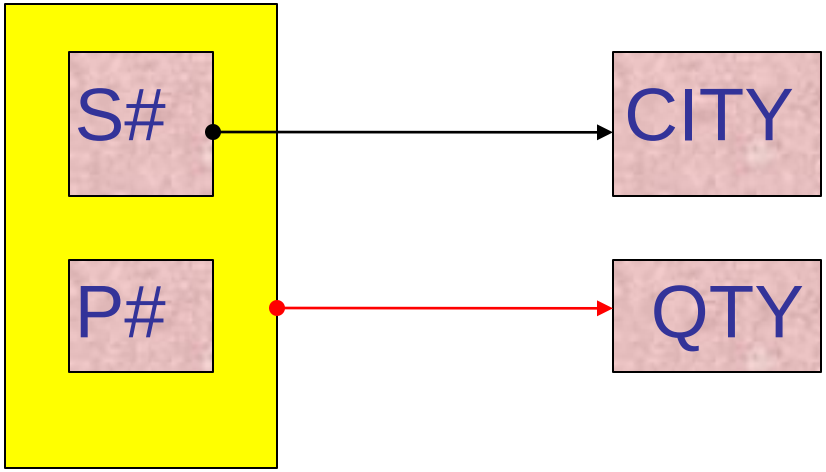
$$A \rightarrow B$$



For the SCP table; its FDs can be represented as:

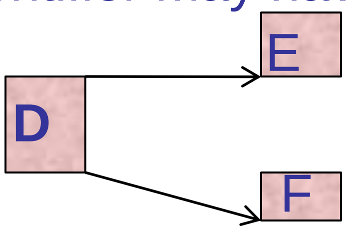
$\{S\#\} \rightarrow \{CITY\}$

$\{S\#, P\#\} \rightarrow \{QTY\}$

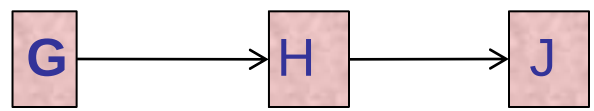


Identifier

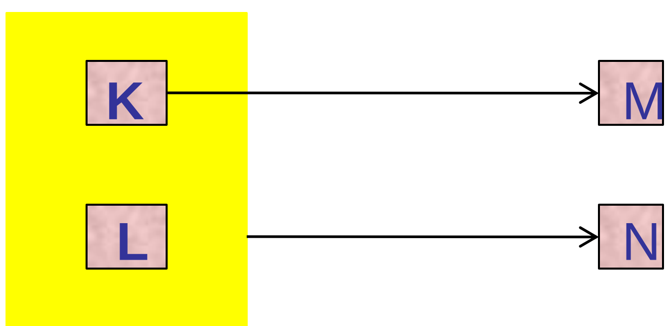
- An identifier is an attribute or composite attributes that can never have duplicate values within a table occurrence, and whose value is sufficient to identify a row. None of the component attributes of an identifier may have NULL values.



a) D is the identifier



b) G is the identifier



c) (K,L) is the identifier