Johns Hopkins Engineering

Principles of Database Systems

Module 5 / Lecture 1

Functional Dependencies and
Normalization for Relational Databases I

JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

Relational Database Design

- Common approaches to database design include top-down and bottom-up.

- Top-down design methodology is a more effective way than bottom-up methodology, as it captures *entity types* with their associated attributes and their *relationships* based on requirements.

Conceptual Database Design

1. Gather and analyze requirements

2. Perform conceptual database design
    - Identify entities and key attributes such as PKs and other candidate keys
    - Identify relationship types with proper cardinalities
    - Identify and associate attributes with entity or relationship types
    - Validate conceptual model against user requirements
    - Review conceptual data model with users and other key stakeholder

- The conceptual design process may be an iterative process.

Good Relational Database Design

- A particular relation *R* is in “good” form to ensure the quality of intension and extension.
  - Logically grouping of attributes can form "good" relation schemas
  - Good logical design leads to good base relations
  - Good design leads good performance and scalability

- A process for reviewing and validating conceptual schema is required.

Informal Design Guidelines for Relational Schema

- Guideline 1:
  - Design a relation schema so that it is easy to explain its meaning. Do not combine attributes from multiple entity types and relationship types into a single relation.
  - Informally, each tuple in a relation should represent an entity instance or relationship instance.

Informal Design Guidelines for Relational Schema (cont.)

- Guideline 1 Example:

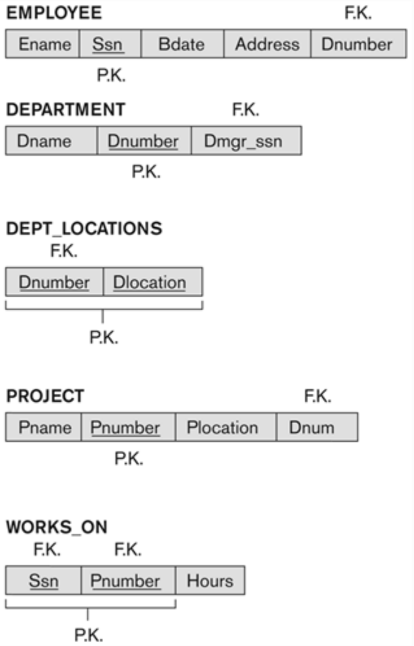


Figure 14.1 A simplified COMPANY relational database schema.

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	291 Berry, 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

Figure 14.2 Sample database state for the relational schema in Figure 14.1.

Johns Hopkins Engineering

Principles of Database Systems

Module 7 / Lecture 2

Functional Dependencies and
Normalization for Relational Databases I

Informal Design Guidelines for Relational Schema (cont.)

- Guideline 2:
  - Design the base relation schemas so that no insertion, deletion, or modification anomalies are present in the relations.

Informal Design Guidelines for Relational Schema (cont.)

- Guideline 2 Example:

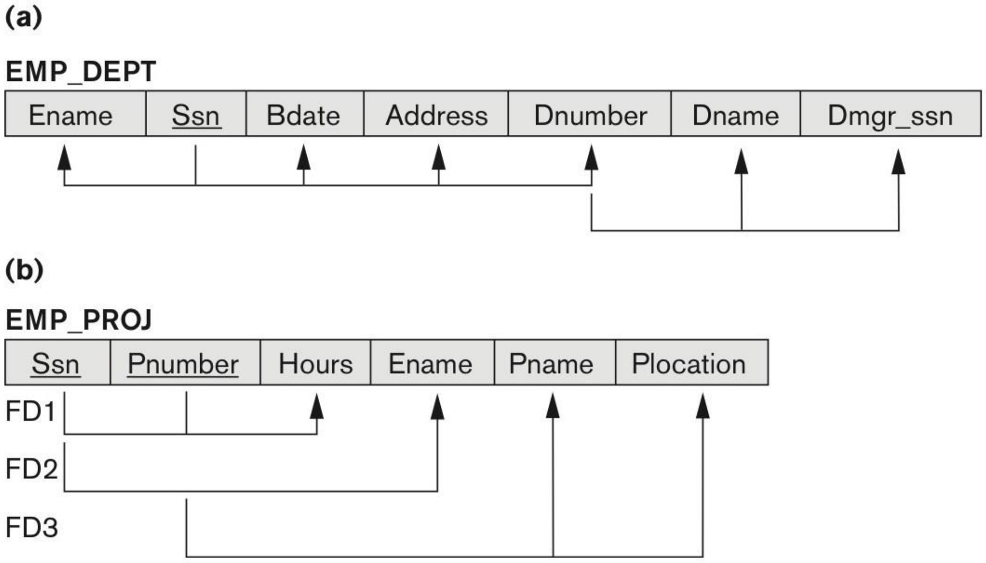


Figure 14.3 Two relation schemas suffering from update anomalies. (a) EMP_DEPT and (b) EMP_PROJ.

Informal Design Guidelines for Relational Schema (cont.)

- Guideline 2 Example:

Redundancy

EMP_DEPT

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

Figure 14.4 Sample states for EMP_DEPT and EMP_PROJ resulting from applying NATURAL JOIN to the relations in Figure 14.2. These may be stored as base relations for performance reasons.

Informal Design Guidelines for Relational Schema (cont.)

- Guideline 2 Example:

For instance, a cluster is a group of tables that share the same data blocks because they share common columns as a cluster key and are often used together.

EMP_PROJ

			Redundancy		Redundancy
Ssn	Pnumber	Hours	Ename	Pname	Plocation
123456789	1	32.5	Smith, John B.	ProductX	Bellaire
123456789	2	7.5	Smith, John B.	ProductY	Sugarland
666884444	3	40.0	Narayan, Ramesh K.	ProductZ	Houston
453453453	1	20.0	English, Joyce A.	ProductX	Bellaire
453453453	2	20.0	English, Joyce A.	ProductY	Sugarland
333445555	2	10.0	Wong, Franklin T.	ProductY	Sugarland
333445555	3	10.0	Wong, Franklin T.	ProductZ	Houston
333445555	10	10.0	Wong, Franklin T.	Computerization	Stafford
333445555	20	10.0	Wong, Franklin T.	Reorganization	Houston
999887777	30	30.0	Zelaya, Alicia J.	Newbenefits	Stafford
999887777	10	10.0	Zelaya, Alicia J.	Computerization	Stafford
987987987	10	35.0	Jabbar, Ahmad V.	Computerization	Stafford
987987987	30	5.0	Jabbar, Ahmad V.	Newbenefits	Stafford
987654321	30	20.0	Wallace, Jennifer S.	Newbenefits	Stafford
987654321	20	15.0	Wallace, Jennifer S.	Reorganization	Houston
888665555	20	Null	Borg, James E.	Reorganization	Houston

Figure 14.4 Sample states for EMP_DEPT and EMP_PROJ resulting from applying NATURAL JOIN to the relations in Figure 14.2. These may be stored as base relations for performance reasons.

Informal Design Guidelines for Relational Schema (cont.)

- Guideline 2 Example:
  - Consider the relation:
    EMP_PROJ (Emp#, Proj#, 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.

    Insert Anomaly: Cannot insert a project unless an employee is assigned to it. Inversely cannot insert an employee unless he/she is assigned to a project.

Informal Design Guidelines for Relational Schema (cont.)

- Guideline 2 Example:
  - 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.

    More storage required.

Informal Design Guidelines for Relational Schema (cont.)

- Guideline 2 Example:
  - **Problems encountered:**
    - More data storage required
    - Insertion anomalies
    - Deletion anomalies
    - Modification anomalies
  - **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**

Informal Design Guidelines for Relational Schema (cont.)

- Guideline 3:
  - If possible, avoid placing attributes in a base relation when their values may frequently be null.
  - Problems encountered:
    - *May* require more data storage
    - *May* not work with aggregate operations (e.g., COUNT, SUM, AVG)
    - *May* have different interpretations

Informal Design Guidelines for Relational Schema (cont.)

- Guideline 4:
  - Design relation schemas so that they can be JOINed with equality conditions on attributes that are either primary keys or foreign keys in a way that guarantees that no spurious tuples are generated.

Informal Design Guidelines for Relational Schema (cont.)

- Guideline 4 Example:

(a)

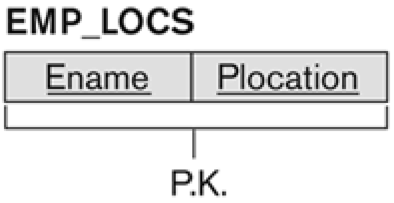


Figure 14.5 Particularly poor design for the EMP_PROJ relation in Figure 14.3(b). (a) The two relation schemas EMP_LOCS and EMP_PROJ1. (b) The result of projecting the extension of EMP_PROJ from Figure 14.4 onto the relations EMP_LOCS and EMP_PROJ1.

EMP_LOCS

Ename	Plocation
Smith, John B.	Bellaire
Smith, John B.	Surgarland
Narayan, Ramesh K.	Houston
English, Joyce A.	Bellaire
English, Joyce A.	Surgarland
Wong, Franklin T.	Surgarland
Wong, Franklin T.	Houston
Wong, Franklin T.	Stafford
Zelaya, Alicia J.	Stafford
Jabbar, Ahmad V.	Stafford
Wallace, Jennifer S.	Stafford
Wallace, Jennifer S.	Houston
Borg, James E.	Houston

Informal Design Guidelines for Relational Schema (cont.)

- Guideline 4 Example:

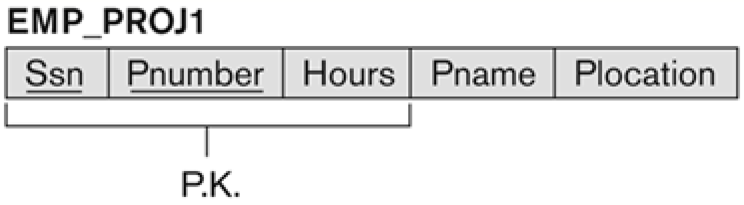


Figure 14.5 Particularly poor design for the EMP_PROJ relation in Figure 14.3(b). (a) The two relation schemas EMP_LOCS and EMP_PROJ1. (b) The result of projecting the extension of EMP_PROJ from Figure 14.4 onto the relations EMP_LOCS and EMP_PROJ1.

EMP_PROJ1

Ssn	Pnumber	Hours	Pname	Plocation
123456789	1	32.5	ProductX	Bellaire
123456789	2	7.5	ProductY	Sugarland
666884444	3	40.0	ProductZ	Houston
453453453	1	20.0	ProductX	Bellaire
453453453	2	20.0	ProductY	Sugarland
333445555	2	10.0	ProductY	Sugarland
333445555	3	10.0	ProductZ	Houston
333445555	10	10.0	Computerization	Stafford
333445555	20	10.0	Reorganization	Houston
999887777	30	30.0	Newbenefits	Stafford
999887777	10	10.0	Computerization	Stafford
987987987	10	35.0	Computerization	Stafford
987987987	30	5.0	Newbenefits	Stafford
987654321	30	20.0	Newbenefits	Stafford
987654321	20	15.0	Reorganization	Houston
888665555	20	NULL	Reorganization	Houston

Informal Design Guidelines for Relational Schema (cont.)

- Guideline 4 Example:

Figure 14.6 Result of applying NATURAL JOIN to the tuples in EMP_PROJ1 and EMP_LOCS of Figure 14.5 just for employee with Ssn = “123456789”. Generated spurious tuples are marked by asterisks.

EMP_LOCS * EMP_PROJ1 (Plocation)

	Ssn	Pnumber	Hours	Pname	Plocation	Ename
	123456789	1	32.5	ProductX	Bellaire	Smith, John B.
*	123456789	1	32.5	ProductX	Bellaire	English, Joyce A.
	123456789	2	7.5	ProductY	Sugarland	Smith, John B.
*	123456789	2	7.5	ProductY	Sugarland	English, Joyce A.
*	123456789	2	7.5	ProductY	Sugarland	Wong, Franklin T.
	666884444	3	40.0	ProductZ	Houston	Narayan, Ramesh K.
*	666884444	3	40.0	ProductZ	Houston	Wong, Franklin T.
*	453453453	1	20.0	ProductX	Bellaire	Smith, John B.
	453453453	1	20.0	ProductX	Bellaire	English, Joyce A.
*	453453453	2	20.0	ProductY	Sugarland	Smith, John B.
	453453453	2	20.0	ProductY	Sugarland	English, Joyce A.
*	453453453	2	20.0	ProductY	Sugarland	Wong, Franklin T.
*	333445555	2	10.0	ProductY	Sugarland	Smith, John B.
*	333445555	2	10.0	ProductY	Sugarland	English, Joyce A.
	333445555	2	10.0	ProductY	Sugarland	Wong, Franklin T.
*	333445555	3	10.0	ProductZ	Houston	Narayan, Ramesh K.
	333445555	3	10.0	ProductZ	Houston	Wong, Franklin T.
	333445555	10	10.0	Computerization	Stafford	Wong, Franklin T.
*	333445555	20	10.0	Reorganization	Houston	Narayan, Ramesh K.
	333445555	20	10.0	Reorganization	Houston	Wong, Franklin T.

*
*
*

Informal Design Guidelines for Relational Schema (cont.)

- Guideline 4:
  - Problems encountered:
    - May create spurious (false or fake) tuples
    - Bad designs for a relational database may result in erroneous results for certain JOIN operations
    - The "lossless join" property is used to guarantee meaningful results for join operations

Johns Hopkins Engineering

Principles of Database Systems

Module 7 / Lecture 3

Functional Dependencies and
Normalization for Relational Databases I

JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

Functional Dependency

- A functional dependency (FD) describes attributes in a relation.

- Notation for Functional Dependency:
  - Let *R* be a relation, *X* and *Y* be arbitrary subsets of the set of attributes of *R*.
  - $X \rightarrow Y$ represents that *X* *functionally determines* *Y*, or *Y* is *functionally dependent on* *X*. If and only if each *X* value in *R* has precisely one *Y* value. In other words, if any two tuples of *R* have the same of *X* value, they also have the same value of *Y*.

Functional Dependency (cont.)

- **For example, the value of a primary key or an alternate key in a relation can functionally determine a subset of other attributes of the relation.**
  - In the EMPLOYEE relation of COMPANY ERD:
    {ssn} → {lname, fname, salary}
    {ssn, salary} → {lname, fname, salary, address}
    {emp_id} → {lname, fname, salary, dnumber}

  - In the STUDENT relation of UNIVSERSITY ERD:
    {student_id} → {lname, fname, bdate, address}

Functional Dependency (cont.)

- Inference Rules for Functional Dependencies

- Six inference rules for functional dependencies
  - Let X, Y, Z and W be arbitrary subsets of the set of attributes of the giver relation R, and let XY be the union of X and Y; Then:

    1. If Y is subset of X, then X → Y (Reflexive Rule)
    2. If X → Y, then XZ → YZ (Augmentation Rule)
    3. If X → Y and Y → Z, then X → Z (Transitive Rule)
    4. If X → YZ, Then X → Y and X → Z (Decomposition Rule)
    5. If X → Y, and X → Z Then X → YZ (Union Rule)
    6. If X → Y, WY → Z, then XW → Z (Pseudo Transitive Rule)

Functional Dependency (cont.)

- Normalization provides a formal framework for analyzing relation schemas based on their keys and on the functional dependencies among their attributes.

- The purpose of normalization is to review and validate relations to establish a proper set of relations that meet the data requirements.

Functional Dependency (cont.)

- **Normal Forms Based on Primary Keys**
  - The process of normalizing a model is one of removing all relational structures that provide multiple ways to know the same fact.
  - Another way to look at normalization is as a method of controlling and eliminating redundancy in data storage.
  - The other objective is to eliminate the update anomalies and lossless joins and ensure there are no spurious tuple problems.

Functional Dependency (cont.)

- **Normal Forms Based on Primary Keys**
  - The goal of normalization is *one fact in one place with one copy* that leads to a good database design principle.
  - Data items belong together in a logical group and a group of items can be identified by its unique identifier.
  - Data in group describes one, and only one, thing.

Type of Keys

- A candidate key *K* for an *R* is a subset of the set of attributes of *R*. It has the following properties:
    - Uniqueness: No two distinct tuples of *R* have the same value for *K*.
    - Irreducibility: No proper subset of *K* has the uniqueness property.

- A primary key (PK) of a relation is a unique identifier for a relation.

Type of Keys (cont.)

- A *superkey* of relation *R* is a set of attributes of *R* that includes at least one candidate key of *R* as a subset.
  For example:
  SSN is a key of EMPLOYEE relation
  {SSN, LNAME, FNAME} is a superkey
  {SSN, LNAME, FNAME, SALARY} is a superkey

- A candidate key is a minimal set of superkey by removing all non-key attributes.

Rolename

- A rolename is a *new* name for a foreign key. A rolename is used to indicate that the domain of the foreign key is a subset of the domain of the attribute in the parent, and performs a specific function (or role) in the entity.

- It is also used when multiple attributes occur in an entity and all are foreign keys from a parent table.
  - Example: Many STATE names may appear multiple times in an EMPLOYEE record or a STUDENT record.
    Rolenames should be assigned to these attributes (e.g., dob_state, current_address_state, permanent_address_state.)

Maintaining Referential Integrity

- A foreign key column value must match an existing primary key column value (or be NULL). Referential integrity constraints are specified on how referential integrity is to be maintained in a schema’s DDL.

- It is necessary to specify a “Delete Constraint” to determine what should happen if a row containing a referenced primary key is deleted.

Maintaining Referential Integrity (cont.)

- It is necessary to specify an "Update Constraint" to determine what should happen if a referenced primary key is updated. This constraint only applies to the situation when the primary key is allowed to be updated.
- Options for Update Constraint and Delete Constraint are CASCADE, RESTRICTED, SET DEFAULT and NULLIFY (only if NULLs are allowed).

Johns Hopkins Engineering

Principles of Database Systems

Module 7 / Lecture 4

Functional Dependencies and
Normalization for Relational Databases I

The Process of Normalization

- Normalization is executed as a series of steps. Each step involves a specific normal form with particular properties.
- Normalization is a process for analyzing relations based on their primary keys and functional dependencies.
- For an Unnormalized Form: A relation contains a repeating group(s) or an attribute contains a set of values.

First Normal Form (1NF)

- A relation R is in 1NF if and only if all underlying domains contain atomic values (single-valued) only, not a set of values, not repeating groups.

- Steps for converting unnormalized form to First Normal Form (1NF):
  - Remove the repeating group from the base table
  - Create a new relation with a proper new PK that includes the PK of the base relation and the repeating group

First Normal Form (1NF) (cont.)

- Examples

(a)

DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocations

(b)

DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocations
Research	5	333445555	{Bellaire, Sugarland, Houston}
Administration	4	987654321	{Stafford}
Headquarters	1	888665555	{Houston}

(c)

DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocation
Research	5	333445555	Bellaire
Research	5	333445555	Sugarland
Research	5	333445555	Houston
Administration	4	987654321	Stafford
Headquarters	1	888665555	Houston

1NF, but not a good design, why?

Another 1NF design

DEPARTMENT

Dnumber	Dname	Dmgr_ssn

DEPT_LOCATION

Dnumber	Dlocation

1NF, why this design is better?

Figure 14.9 Normalization into 1NF. (a) A relation schema that is not in 1NF. (b) Sample state of relation DEPARTMENT. (c) 1NF version of the same relation with redundancy.

Second Normal Form (2NF)

- A relation R is in 2NF if it is in 1NF and every non-key attribute is fully functionally dependent on the primary key (no partial dependencies).

- Every non-key attribute must be dependent upon *all parts* of the primary key of a relation.

- If each column is not dependent upon the *entire* primary key, the relation is not in 2NF.

Second Normal Form (2NF) (cont.)

- Steps for converting 2NF:
  - Determine which non-key attributes do not depend on the relation’s *entire* primary key
  - Remove those attributes from the base relation
  - Create a second relation with those attributes and the attribute(s) from the PK that they are dependent upon

- Any single attribute (associated as a primary key) relation is automatically in 2NF.

Second Normal Form (2NF) (cont.)

- Example

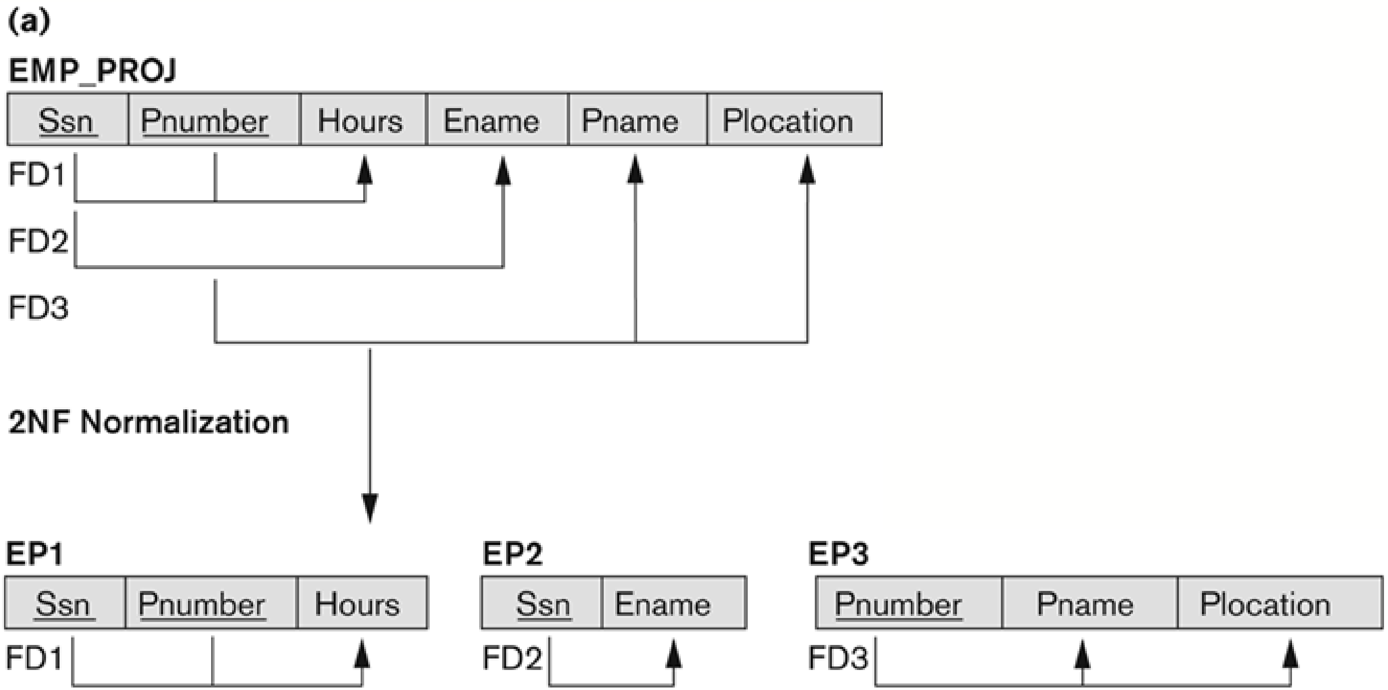


Figure 14.11 Normalizing into 2NF. (a) Normalizing EMP_PROJ into 2NF relations.

Third Normal Form (3NF)

- A relation R is in 3NF if it is in 2NF and no non-key attribute of R is functionally dependent on another non-key attribute.

- A relation R is in 3NF if it is in 2NF and every non-key attribute is non-transitively dependent on the primary key.
    - Transitive dependency

Third Normal Form (3NF) (cont.)

- Steps for converting 3NF:
  - Determine which attributes are dependent upon another non-key attribute
  - Remove those attributes from the base relation
  - Create a second relation with those attributes and the non-key attribute that they are dependent upon

Third Normal Form (3NF) (cont.)

- Example: Please describe EMP_DEPT FDs (<u>Ssn</u>)

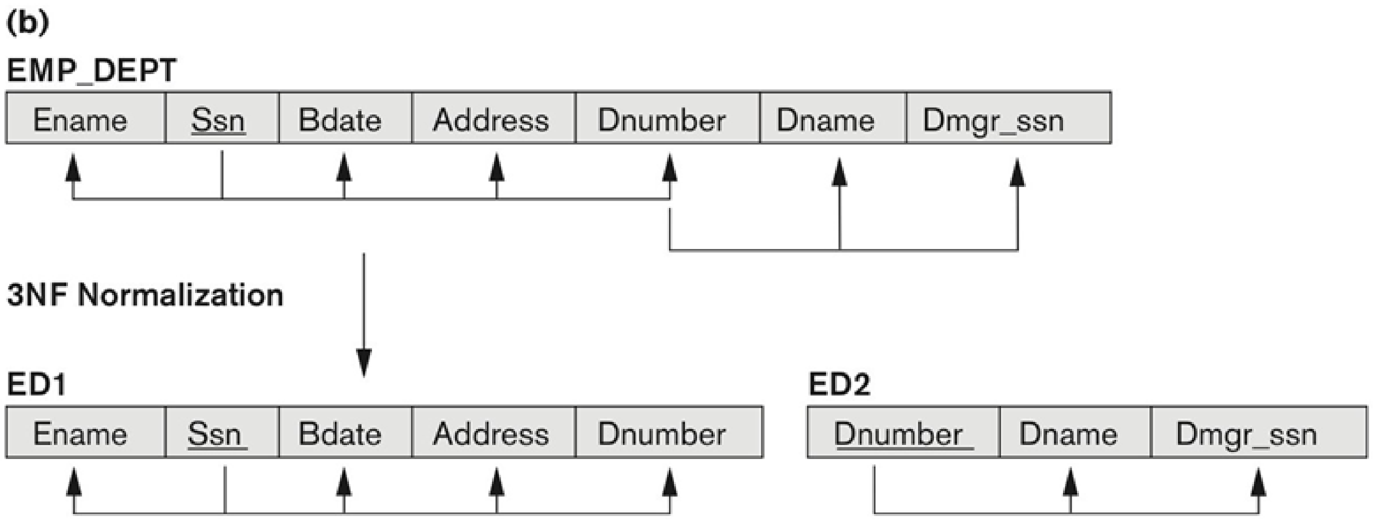


Figure 14.11 Normalizing into 3NF. (b) Normalizing EMP_DEPT into 3NF relations.

Johns Hopkins Engineering

Principles of Database Systems

Module 7 / Lecture 5

Functional Dependencies and
Normalization for Relational Databases I

Normalization Exercise

- An ORDER Relation – an unnormalized form

ORDER

order_ id	order_ date	customer_ id	customer_ name	customer_ address	product_ id	product_ name	product_ price	quantity_ ordered	product_ total	order_ total

|______|
PK

(This order is an original entity. One order may have many products. There is a repeating group, the grey attributes, for product information.)

Normalizing into First Normal Form (1NF):

Normalizing into Second Normal Form (2NF):

Normalizing into Third Normal Form (3NF):

Normalization Exercise (cont.)

- First Normal Form (1NF)
  - Steps for converting 1NF:
    1. Remove the repeating group from the base table.
    2. Create a new table with the PK of the base table and the repeating group

ORDER

order_ id	order_ date	customer_ id	customer_ name	customer_ address	order_ total

PK

order_id → {order_date, customer_id, customer_name, customer_address, order_total}

ORDER_PRODUCT

order_ id	product_ id	product_ name	product_ price	quantity_ ordered	product_ total

PK {order_id, product_id} → {product_name, product_price, quantity_ordered, product_total}

Normalization Exercise (cont.)

- Second Normal Form (2NF)
  - Steps for converting 2NF:
    1. Determine which non-key columns are not dependent upon the table's entire primary key.
    2. Remove those columns from the base table.
    3. Create a second table with those columns and the column(s) from the PK that they are dependent upon.

ORDER

order_ id	order_ date	customer_ id	customer_ name	customer_ address	order_ total

PK

ORDER_PRODUCT

order_ id	product_ id	quantity_ ordered	product_ total

PK

{order_id, product_id} → {quantity_ordered, product_total}

PRODUCT

product_ id	product_ name	product_ price

PK

product_id → {product_name, product_price}

Normalization Exercise (cont.)

- Third Normal Form (3NF)
  - Steps for converting 3NF:
    1. Determine which columns are dependent upon another non-key column.
    2. Remove those columns from the base table.
    3. Create a second table with those columns and the non-key column that they are dependent upon.

ORDER

order_ id	order_ date	customer_ id	order_ total

PK: order_id

ORDER_PRODUCT

order_ id	product_ id	quantity_ ordered	product_ total

PK: order_id, product_id

CUSTOMER

customer_ id	customer_ name	customer_ address

PK: customer_id

customer_id → {customer_name, customer_address}

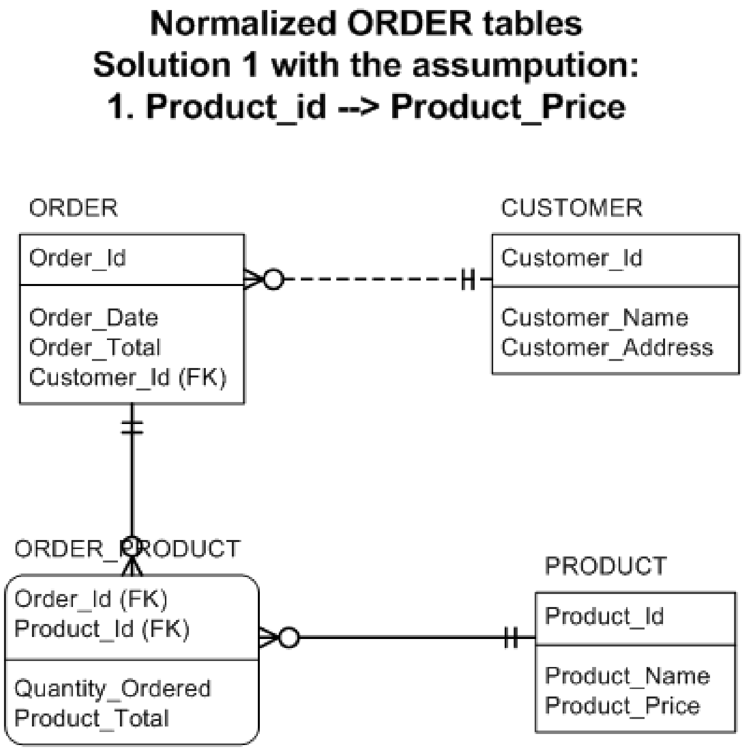
PRODUCT

product_ id	product_ name	product_ price

PK: product_id

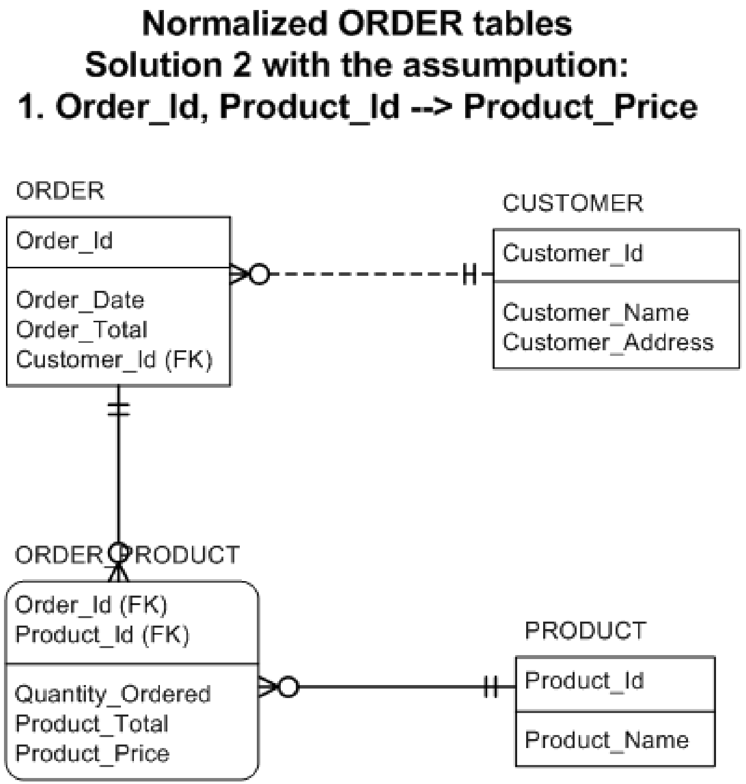
Normalization Exercise (cont.)

- Normalized ORDER in 3NF
  - Reverse the normalized relations to an ERD
    - Product_id → Product_Price
    - Product_Price: a current product price associated with a product instance



Normalization Exercise (cont.)

- Normalized ORDER in 3NF
  - Reverse the normalized relations to an ERD
    - {Order_id, Product_id} → Product_Price
    - Product_Price: a price at the time of an order transaction



Basic Principles When Performing Normalization

- No additional attributes will be added during the normalization process.
- It is important that you do the 1NF, 2NF, and 3NF in sequence. Otherwise, you may get the wrong answer.

Summary of Normal Forms Based on Primary Keys and Corresponding Normalization

Table 14.1 Summary of Normal Forms Based on Primary Keys and Corresponding Normalization

Normal Form	Test	Remedy (Normalization)
First (1NF)	Relation should have no multivalued attributes or nested relations.	Form new relations for each multivalued attribute or nested relation.
Second (2NF)	For relations where primary key contains multiple attributes, no nonkey attribute should be functionally dependent on a part of the primary key.	Decompose and set up a new relation for each partial key with its dependent attribute(s). Make sure to keep a relation with the original primary key and any attributes that are fully functionally dependent on it.
Third (3NF)	Relation should not have a nonkey attribute functionally determined by another nonkey attribute (or by a set of nonkey attributes). That is, there should be no transitive dependency of a nonkey attribute on the primary key.	Decompose and set up a relation that includes the nonkey attribute(s) that functionally determine(s) other nonkey attribute(s).

Boyce/Codd Normal Form (BCNF)

- A BCNF is stricter than 3NF.
- A relation is in BCNF if and only if every determinant in the relation is a candidate key.
- When a relation contains only one candidate key, the 3NF and the BCNF are equivalent.
- If a relation has more than one candidate key, the relation may require further normalization.

Boyce/Codd Normal Form (BCNF) (cont.)

<u>A</u>	<u>B</u>	C	D

FD 1: {A, B} → {C, D}
FD 2: C → B
3NF but not in BCNF

<u>A</u>	<u>C</u>	D

<u>C</u>	B

<u>Student_id</u>	<u>Faculty_id</u>	Offering_id	Grade

FD 1: {Student_id, Faculty_id} → {Offering_id, Grade}
FD 2: Offering_id → Faculty_id
3NF but not in BCNF (with some assumptions)

<u>Student_id</u>	<u>Offering_id</u>	Grade

<u>Offering_id</u>	Faculty_id