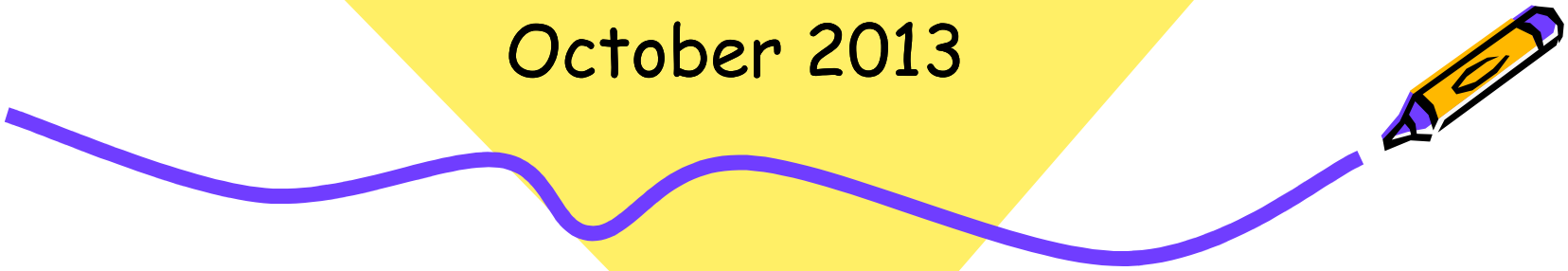
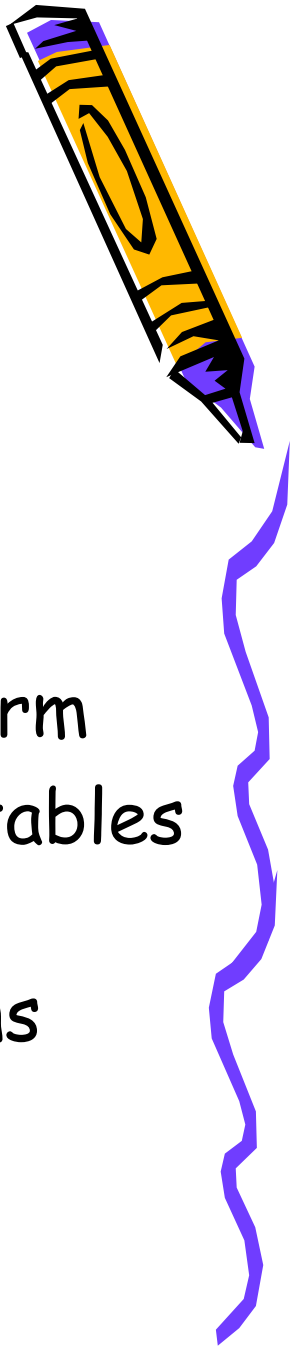
RELATIONAL MODEL & NORMALIZATION

DAT-BAS

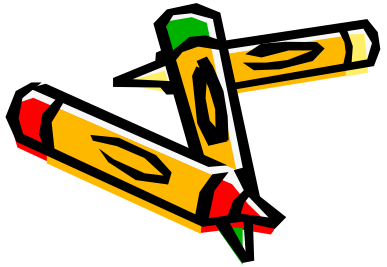
October 2013



Objectives

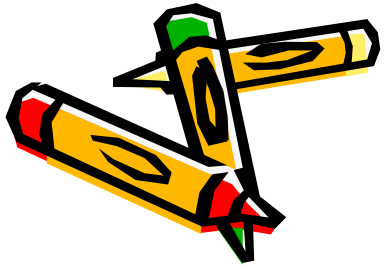


- Definition of terms
- List properties of relations
- State two properties of candidate keys
- Define first, second, and third normal form
- Use normalization to convert anomalous tables to well-structured relations
- Describe problems from merging relations

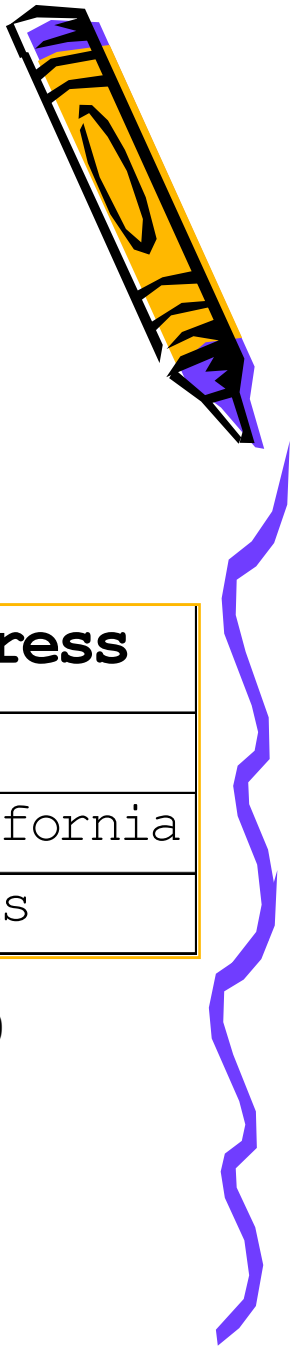


What is a RELATION?

- A structure (two-dimensional table) with columns and rows
  - Columns → Attributes
  - Domain → Allowable values for one or more attributes
  - Rows → Tuple
  - Degree → No. of Attributes of a Relation
  - Cardinality → No. of Tuples of a Relation



Sample Relation



Relational Schema

Relation Name

Tuples

Attributes

Cardinality

Degree

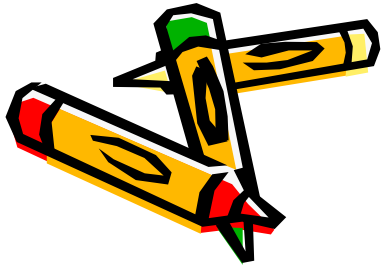
Domain of Idno

Person (idno, fname, lname, address)

Person

idno	fname	lname	address
95023	Milan	Milenkovic	NY
95924	C. J.	Date	California
95025	Ramez	Elmasri	Texas

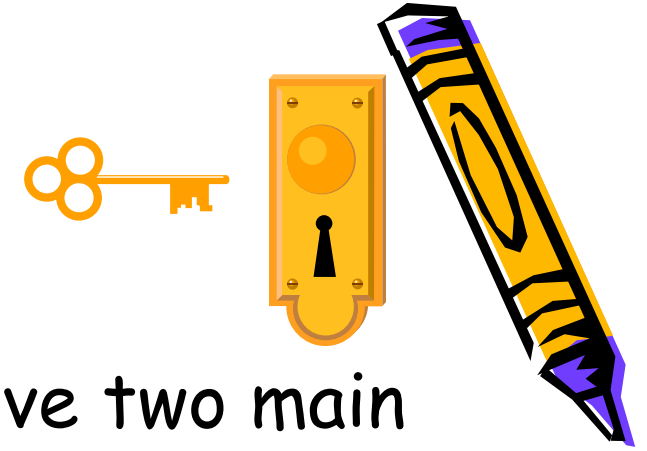
5 digit integer starting from 00001-99999



Properties of a Relation

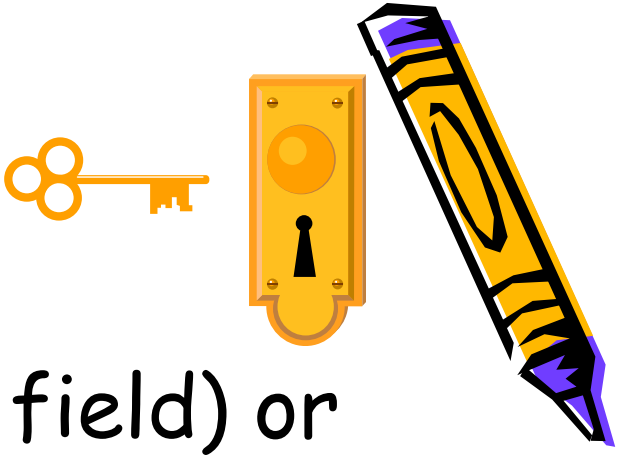
- Requirements for a table to qualify as a relation:
  - It must have a unique name.
  - Every attribute value must be atomic (not multi-valued, not composite)
  - Every tuple (row) must be unique (can't have two tuples with exactly the same values for all their fields)
  - Attributes (columns) in tables must have unique names
  - The order of the columns must be irrelevant
  - The order of the tuples must be irrelevant

Key Fields

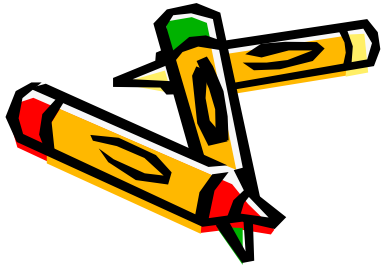


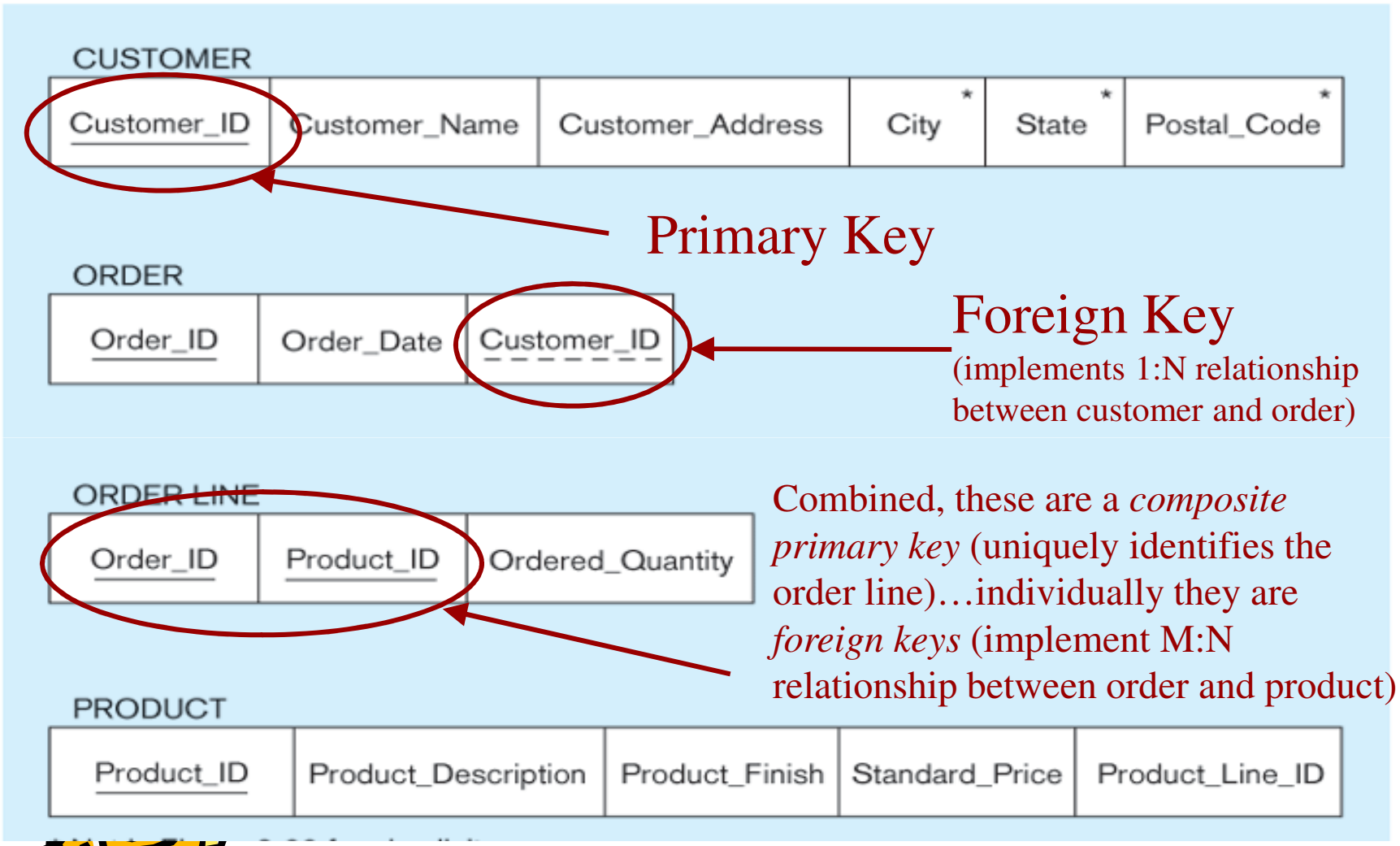
- Keys are special fields that serve two main purposes:
  - ***Primary keys*** are unique identifiers of the relation in question. Examples include employee numbers, social security numbers, etc. *This is how we can guarantee that all rows are unique*
  - ***Foreign keys*** are identifiers that enable a dependent relation (on the many side of a relationship) to refer to its parent relation (on the one side of the relationship)

Key Fields



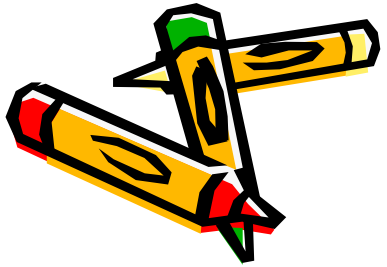
- Keys can be ***simple*** (a single field) or ***composite*** (more than one field)
- Keys usually are used as indexes to speed up the response to user queries




CUSTOMER
Customer_ID
Customer_Name
Customer_Address
City *
State *
Postal_Code *
Primary Key
ORDER
Order_ID
Order_Date
Customer_ID
Foreign Key
(implements 1:N relationship between customer and order)
ORDER LINE
Order_ID
Product_ID
Ordered_Quantity
Combined, these are a composite primary key (uniquely identifies the order line)…individually they are foreign keys (implement M:N relationship between order and product)
PRODUCT
Product_ID
Product_Description
Product_Finish
Standard_Price
Product_Line_ID

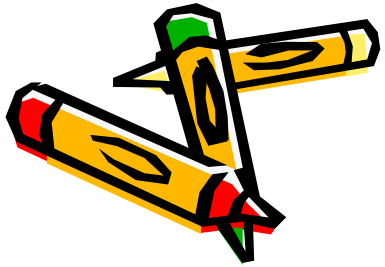
Integrity Constraints

- Domain Constraints
  - Allowable values for an attribute.
- Entity Integrity
  - No primary key attribute may be null. All primary key fields **MUST** have data



Domain Definition for INVOICE attributes

Attribute	*Domain Name*	*Description*	*Domain*
Customer_ID	Customer_IDs	Set of all possible customer IDs	character: size 5
Customer_Name	Customer_Names	Set of all possible customer names	character: size 25
Customer_Address	Customer_Addresses	Set of all possible customer addresses	character: size 30
City	Cities	Set of all possible cities	character: size 20
State	States	Set of all possible states	character: size 2
Postal_Code	Postal_Codes	Set of all possible postal zip codes	character: size 10
Order_ID	Order_IDs	Set of all possible order IDs	character: size 5
Order_Date	Order_Dates	Set of all possible order dates	date format mm/dd/yy
Product_ID	Product_IDs	Set of all possible product IDs	character: size 5
Product_Description	Product_Descriptions	Set of all possible product descriptions	character size 25
Product_Finish	Product_Finishes	Set of all possible product finishes	character: size 15
Standard_Price	Unit_Prices	Set of all possible unit prices	monetary: 6 digits
Product_Line_ID	Product_Line_IDs	Set of all possible product line IDs	integer: 3 digits
Ordered_Quantity	Quantities	Set of all possible ordered quantities	integer: 3 digits

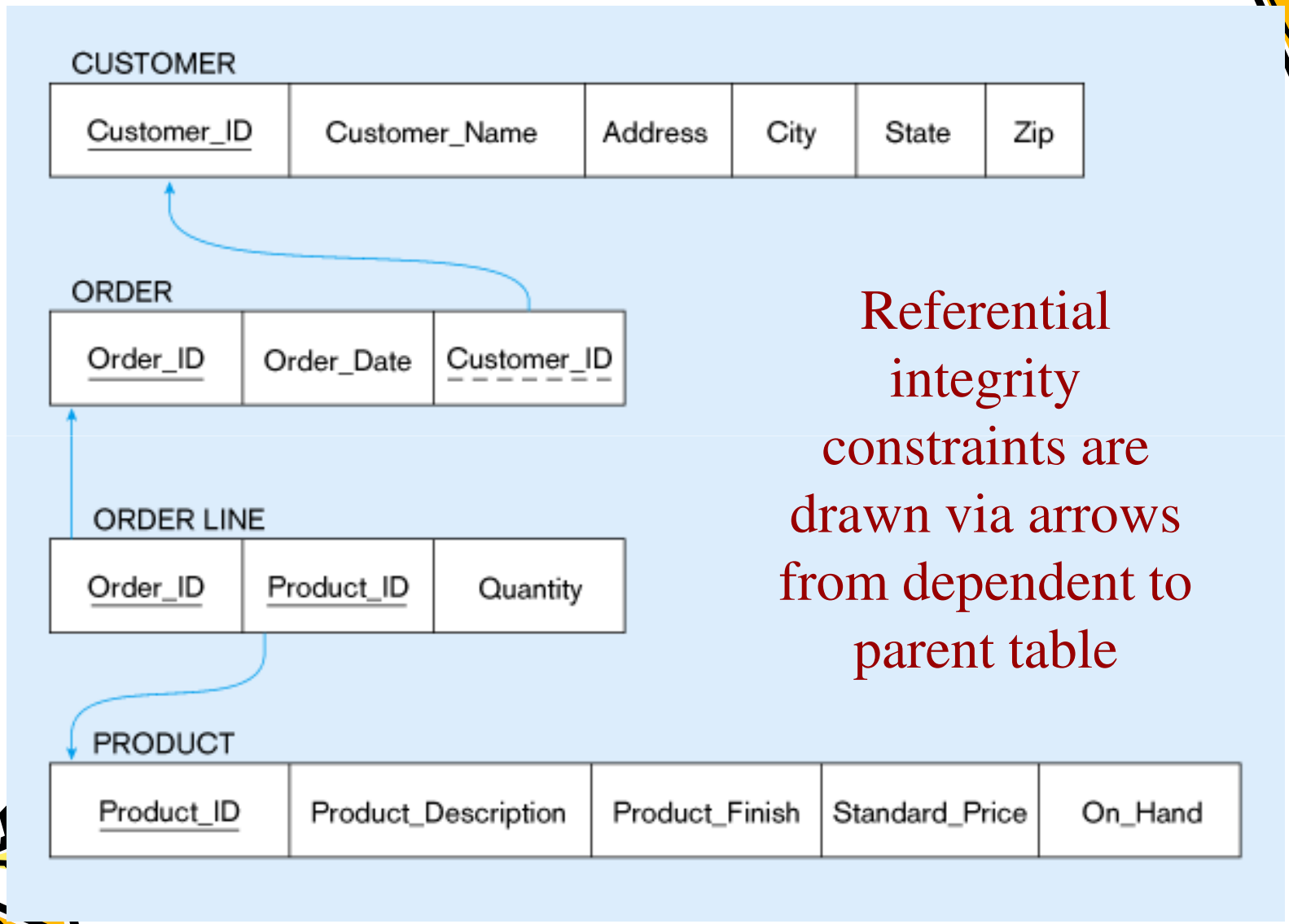


Domain definitions enforce domain integrity constraints

Integrity Constraints

- Referential Integrity – states that any foreign key value (on the relation of the many side) MUST match a primary key value in the relation of the one side (or the foreign key can be null).
  - For example: Delete Rules
    - Restrict – don't allow delete of "parent" side if related rows exist in "dependent" side
    - Cascade – automatically delete "dependent" side rows that correspond with the "parent" side row to be deleted
    - Set-to-Null – set the foreign key in the dependent side to null if deleting from the parent side → not allowed for weak entities

Referential integrity constraints (Pine Valley Furniture)



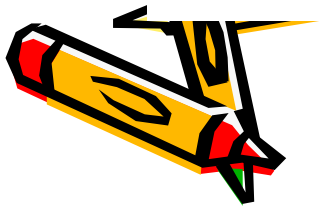
Referential integrity constraints are drawn via arrows from dependent to parent table

```
CREATE TABLE CUSTOMER
        (CUSTOMER_ID                VARCHAR(5)          NOT NULL,
        CUSTOMER_NAME               VARCHAR(25)         NOT NULL,
        CUSTOMER ADDRESS            VARCHAR(30)         NOT NULL,
        CITY                        VARCHAR(20)         NOT NULL,
        STATE                       CHAR(2)             NOT NULL,
        POSTAL_CODE                 CHAR(10)            NOT NULL,
PRIMARY KEY (CUSTOMER_ID);

CREATE TABLE ORDER
        (ORDER_ID                   CHAR(5)             NOT NULL,
        ORDER DATE                  DATE                NOT NULL,
        CUSTOMER_ID                 VARCHAR(5)          NOT NULL,
PRIMARY KEY (ORDER_ID),
FOREIGN KEY (CUSTOMER_ID) REFERENCES CUSTOMER (CUSTOMER_ID);

CREATE TABLE ORDER_LINE
        (ORDER_ID                   CHAR(5)             NOT NULL,
        PRODUCT_ID                  CHAR(5)             NOT NULL,
        ORDERED_QUANTITY            INT                 NOT NULL,
PRIMARY KEY (ORDER_ID, PRODUCT_ID),
FOREIGN KEY (ORDER_ID) REFERENCES ORDER (ORDER_ID),
FOREIGN KEY (PRODUCT_ID) REFERENCES PRODUCT (PRODUCT_ID);

CREATE TABLE PRODUCT
        (PRODUCT_ID                 CHAR(5)             NOT NULL,
        PRODUCT_DESCRIPTION         VARCHAR(25),
        PRODUCT_FINISH              VARCHAR(12),
        STANDARD_PRICE              DECIMAL(8,2)        NOT NULL,
        PRODUCT_LINE_ID             INT                 NOT NULL,
PRIMARY KEY (PRODUCT_ID);
```



Referential integrity constraints are implemented with foreign key to primary key references

What is a Relational Database?

- A Collection of normalized Relations!

  An example relation database schema is given below:

  FACULTY (fidno, fname, lname, address, gender, dob, department, stat)

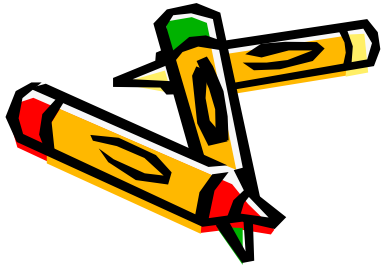
  STUDENT (sidno, fname, lname, address, gender, dob, specialization)

  COURSE (ccode, cname, department, no_of_units)

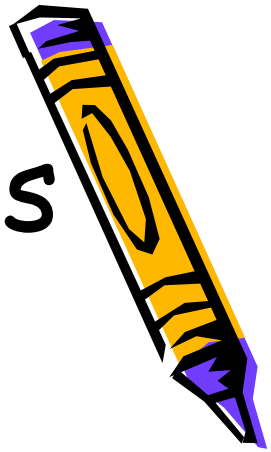
  ENROLLMENT (sidno, ccode, section, day, time, room, fidno)

  PLANTILLA (fidno, ccode, term, SY, section)

  TRANSCRIPT (sidno, ccode, term, SY, section, GPA)

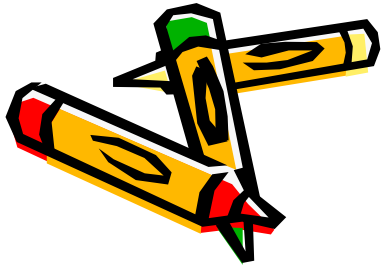


Well-Structured Relations

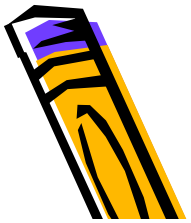


- A relation that contains minimal data redundancy and allows users to insert, delete, and update rows without causing data inconsistencies
- Goal is to avoid anomalies
  - **Insertion Anomaly** – adding new rows forces user to create duplicate data
  - **Deletion Anomaly** – deleting rows may cause a loss of data that would be needed for other future rows
  - **Modification Anomaly** – changing data in a row forces changes to other rows because of duplication

General rule of thumb: a table should not pertain to more than one entity type



Example



EMPLOYEE2

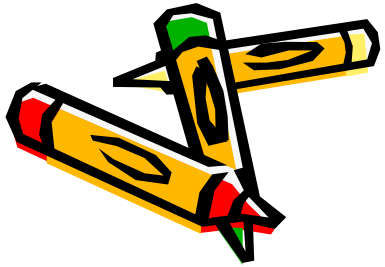
Emp_ID	Name	Dept_Name	Salary	Course_Title	Date_Completed
100	Margaret Simpson	Marketing	48,000	SPSS	6/19/200X
100	Margaret Simpson	Marketing	48,000	Surveys	10/7/200X
140	Alan Beeton	Accounting	52,000	Tax Acc	12/8/200X
110	Chris Lucero	Info Systems	43,000	SPSS	1/12/200X
110	Chris Lucero	Info Systems	43,000	C++	4/22/200X
190	Lorenzo Davis	Finance	55,000		
150	Susan Martin	Marketing	42,000	SPSS	6/19/200X
150	Susan Martin	Marketing	42,000	Java	8/12/200X

Question – Is this a relation?

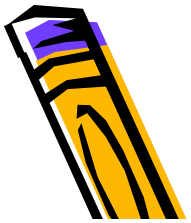
Answer – Yes: unique rows and no multi-valued attributes

Question – What's the primary key?

Answer – Composite: Emp_ID, Course_Title



Anomalies in this Table

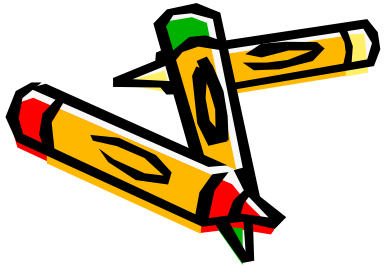


EMPLOYEE2

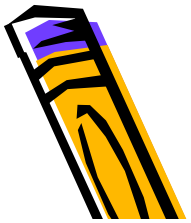
Emp_ID	Name	Dept_Name	Salary	Course_Title	Date_Completed
100	Margaret Simpson	Marketing	48,000	SPSS	6/19/200X
100	Margaret Simpson	Marketing	48,000	Surveys	10/7/200X
140	Alan Beeton	Accounting	52,000	Tax Acc	12/8/200X
110	Chris Lucero	Info Systems	43,000	SPSS	1/12/200X
110	Chris Lucero	Info Systems	43,000	C++	4/22/200X
190	Lorenzo Davis	Finance	55,000		
150	Susan Martin	Marketing	42,000	SPSS	6/19/200X
150	Susan Martin	Marketing	42,000	Java	8/12/200X

A new employee was hired. Can it be added to the DB?

- We can't enter a new employee without having the employee take a class



Anomalies in this Table

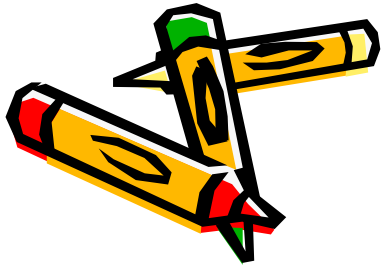


EMPLOYEE2

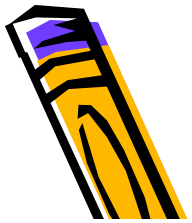
Emp_ID	Name	Dept_Name	Salary	Course_Title	Date_Completed
100	Margaret Simpson	Marketing	48,000	SPSS	6/19/200X
100	Margaret Simpson	Marketing	48,000	Surveys	10/7/200X
140	Alan Beeton	Accounting	52,000	Tax Acc	12/8/200X
110	Chris Lucero	Info Systems	43,000	SPSS	1/12/200X
110	Chris Lucero	Info Systems	43,000	C++	4/22/200X
190	Lorenzo Davis	Finance	55,000		
150	Susan Martin	Marketing	42,000	SPSS	6/19/200X
150	Susan Martin	Marketing	42,000	Java	8/12/200X

Alan Beeton resigned and his record was deleted. What happens?

- Information about the existence of a Tax Acc class will be lost.



Anomalies in this Table



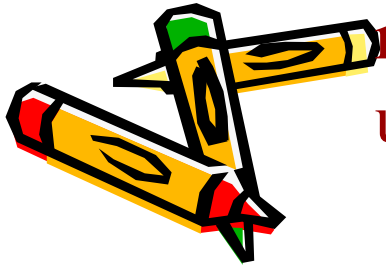
EMPLOYEE2

Emp_ID	Name	Dept_Name	Salary	Course_Title	Date_Completed
100	Margaret Simpson	Marketing	48,000	SPSS	6/19/200X
100	Margaret Simpson	Marketing	48,000	Surveys	10/7/200X
140	Alan Beeton	Accounting	52,000	Tax Acc	12/8/200X
110	Chris Lucero	Info Systems	43,000	SPSS	1/12/200X
110	Chris Lucero	Info Systems	43,000	C++	4/22/200X
190	Lorenzo Davis	Finance	55,000		
150	Susan Martin	Marketing	42,000	SPSS	6/19/200X
150	Susan Martin	Marketing	42,000	Java	8/12/200X

- **Modification** – giving a salary increase to employee 100 forces us to update multiple records

Why do these anomalies exist?

Because there are two themes (entity types) into one relation. This results in duplication, and an unnecessary dependency between the entities

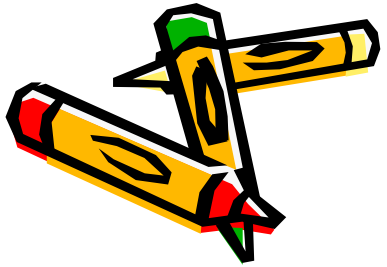


Data Normalization

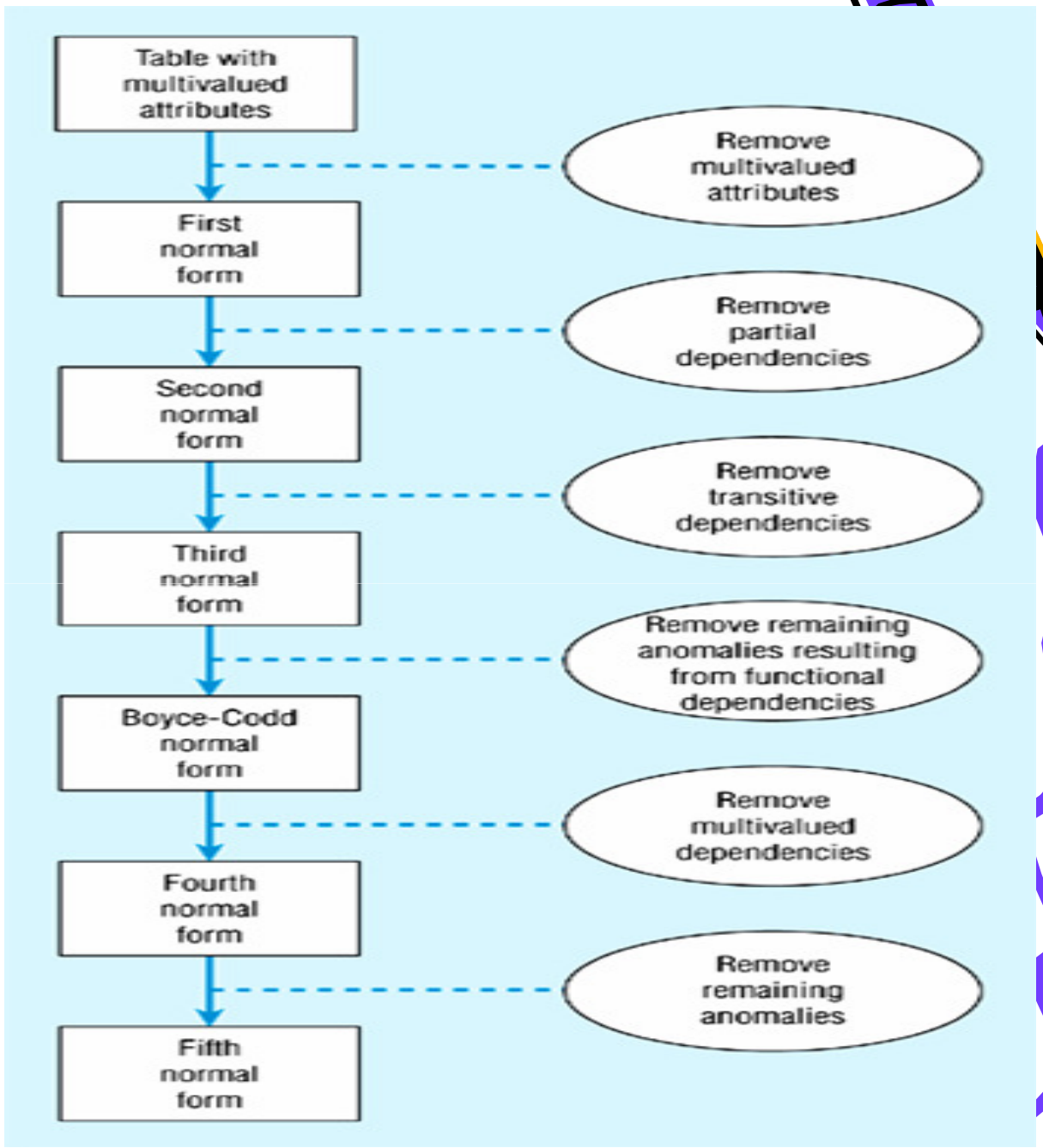
- Primarily a tool to validate and improve a logical design so that it satisfies certain constraints that ***avoid unnecessary duplication of data***
- The process of decomposing relations with anomalies to produce smaller, ***well-structured*** relations

Functional Dependencies and Keys

- Functional Dependency: The value of one attribute (the ***determinant***) determines the value of another attribute
- Candidate Key:
  - A unique identifier. One of the candidate keys will become the primary key
    - E.g. perhaps there is both credit card number and SSN (which are unique for each employee) in a table…in this case both are candidate keys
  - Each non-key field is functionally dependent on every candidate key



Steps in normalization



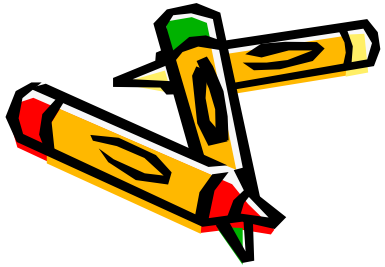
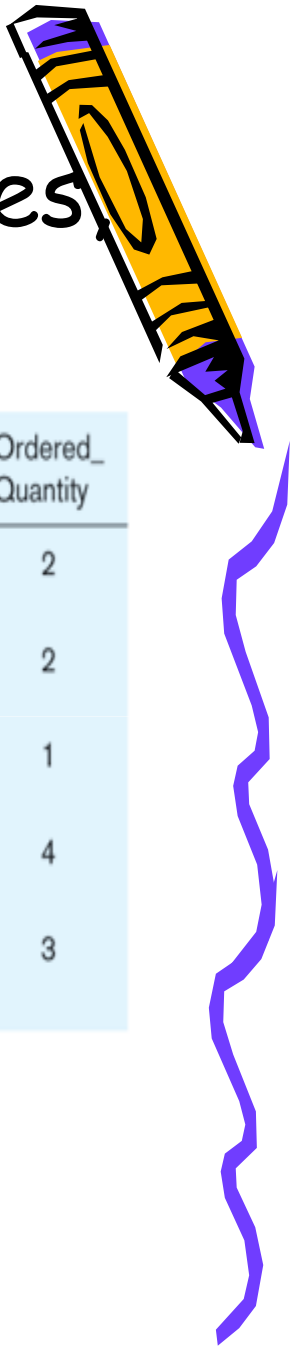
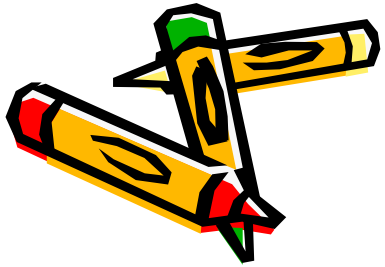


Table with multi-valued attributes, not in 1st normal form



Order_ID	Order_ Date	Customer_ ID	Customer_ Name	Customer_ Address	Product_ID	Product_ Description	Product_ Finish	Unit_ Price	Ordered_ Quantity
1006	10/24/2004	2	Value Furniture	Plano, TX	7	Dining Table	Natural Ash	800.00	2
					5	Writer's Desk	Cherry	325.00	2
					4	Entertainment Center	Natural Maple	650.00	1
1007	10/25/2004	6	Furniture Gallery	Boulder, CO	11	4–Dr Dresser	Oak	500.00	4
					4	Entertainment Center	Natural Maple	650.00	3



Note: this is NOT a relation

First Normal Form

- No multi-valued attributes
- Every attribute value is atomic
- ***All relations*** **are in 1st Normal Form**

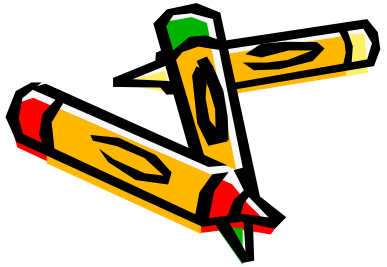


Table with no multi-valued attributes and unique rows, in 1st normal form

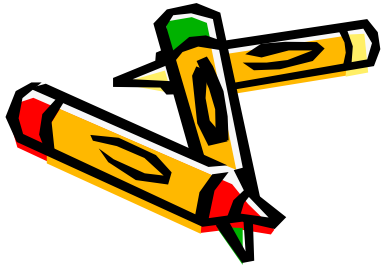
Order_ID	Order_ Date	Customer_ ID	Customer_ Name	Customer_ Address	Product_ID	Product_ Description	Product_ Finish	Unit_ Price	Ordered_ Quantity
1006	10/24/2004	2	Value Furniture	Plano, TX	7	Dining Table	Natural Ash	800.00	2
1006	10/24/2004	2	Value Furniture	Plano, TX	5	Writer's Desk	Cherry	325.00	2
1006	10/24/2004	2	Value Furniture	Plano, TX	4	Entertainment Center	Natural Maple	650.00	1
1007	10/25/2004	6	Furniture Gallery	Boulder, CO	11	4-Dr Dresser	Oak	500.00	4
1007	10/25/2004	6	Furniture Gallery	Boulder, CO	4	Entertainment Center	Natural Maple	650.00	3

Note: this is relation, but not a well-structured one

Anomalies in this Table

Order_ID	Order_ Date	Customer_ ID	Customer_ Name	Customer_ Address	Product_ID	Product_ Description	Product_ Finish	Unit_ Price	Ordered_ Quantity
1006	10/24/2004	2	Value Furniture	Plano, TX	7	Dining Table	Natural Ash	800.00	2
1006	10/24/2004	2	Value Furniture	Plano, TX	5	Writer's Desk	Cherry	325.00	2
1006	10/24/2004	2	Value Furniture	Plano, TX	4	Entertainment Center	Natural Maple	650.00	1
1007	10/25/2004	6	Furniture Gallery	Boulder, CO	11	4-Dr Dresser	Oak	500.00	4
1007	10/25/2004	6	Furniture Gallery	Boulder, CO	4	Entertainment Center	Natural Maple	650.00	3

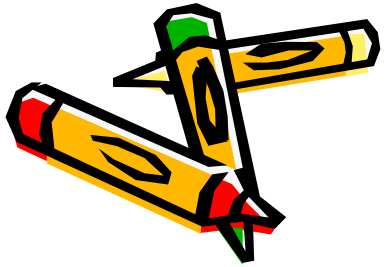
Insertion – if new product is ordered by 1007, customer data must be re-entered, causing duplication



Anomalies in this Table

Order_ID	Order_ Date	Customer_ ID	Customer_ Name	Customer_ Address	Product_ID	Product_ Description	Product_ Finish	Unit_ Price	Ordered_ Quantity
1006	10/24/2004	2	Value Furniture	Plano, TX	7	Dining Table	Natural Ash	800.00	2
1006	10/24/2004	2	Value Furniture	Plano, TX	5	Writer's Desk	Cherry	325.00	2
1006	10/24/2004	2	Value Furniture	Plano, TX	4	Entertainment Center	Natural Maple	650.00	1
1007	10/25/2004	6	Furniture Gallery	Boulder, CO	11	4-Dr Dresser	Oak	500.00	4
1007	10/25/2004	6	Furniture Gallery	Boulder, CO	4	Entertainment Center	Natural Maple	650.00	3

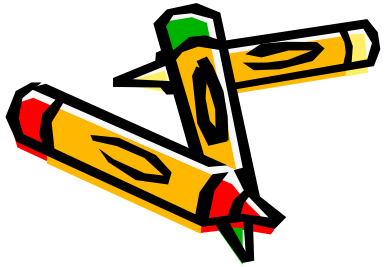
Deletion – if we delete the Dining Table from Order 1006, we lose information concerning this item's finish and price



Anomalies in this Table

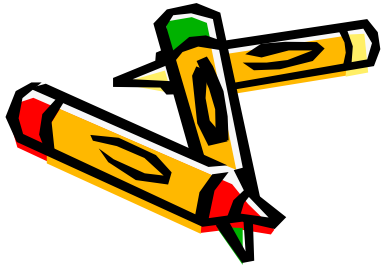
Order_ID	Order_ Date	Customer_ ID	Customer_ Name	Customer_ Address	Product_ID	Product_ Description	Product_ Finish	Unit_ Price	Ordered_ Quantity
1006	10/24/2004	2	Value Furniture	Plano, TX	7	Dining Table	Natural Ash	800.00	2
1006	10/24/2004	2	Value Furniture	Plano, TX	5	Writer's Desk	Cherry	325.00	2
1006	10/24/2004	2	Value Furniture	Plano, TX	4	Entertainment Center	Natural Maple	650.00	1
1007	10/25/2004	6	Furniture Gallery	Boulder, CO	11	4-Dr Dresser	Oak	500.00	4
1007	10/25/2004	6	Furniture Gallery	Boulder, CO	4	Entertainment Center	Natural Maple	650.00	3

Update – changing the price of product ID 4 requires update in several records

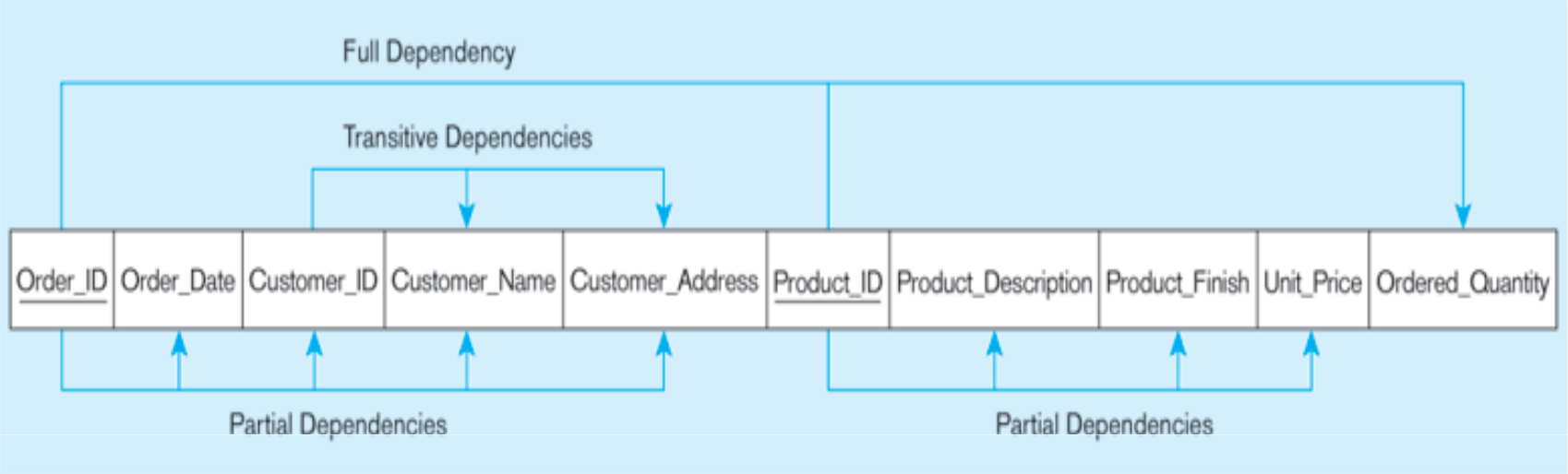


Second Normal Form

- 1NF PLUS ***every non-key attribute is fully functionally dependent on the ENTIRE primary key***
  - Every non-key attribute must be defined by the entire key, not by only part of the key
  - No partial functional dependencies



Sample Functional Dependency Diagram for an INVOICE



Order_ID ➔ Order_Date, Customer_ID, Customer_Name, Customer_Address

Customer_ID ➔ Customer_Name, Customer_Address

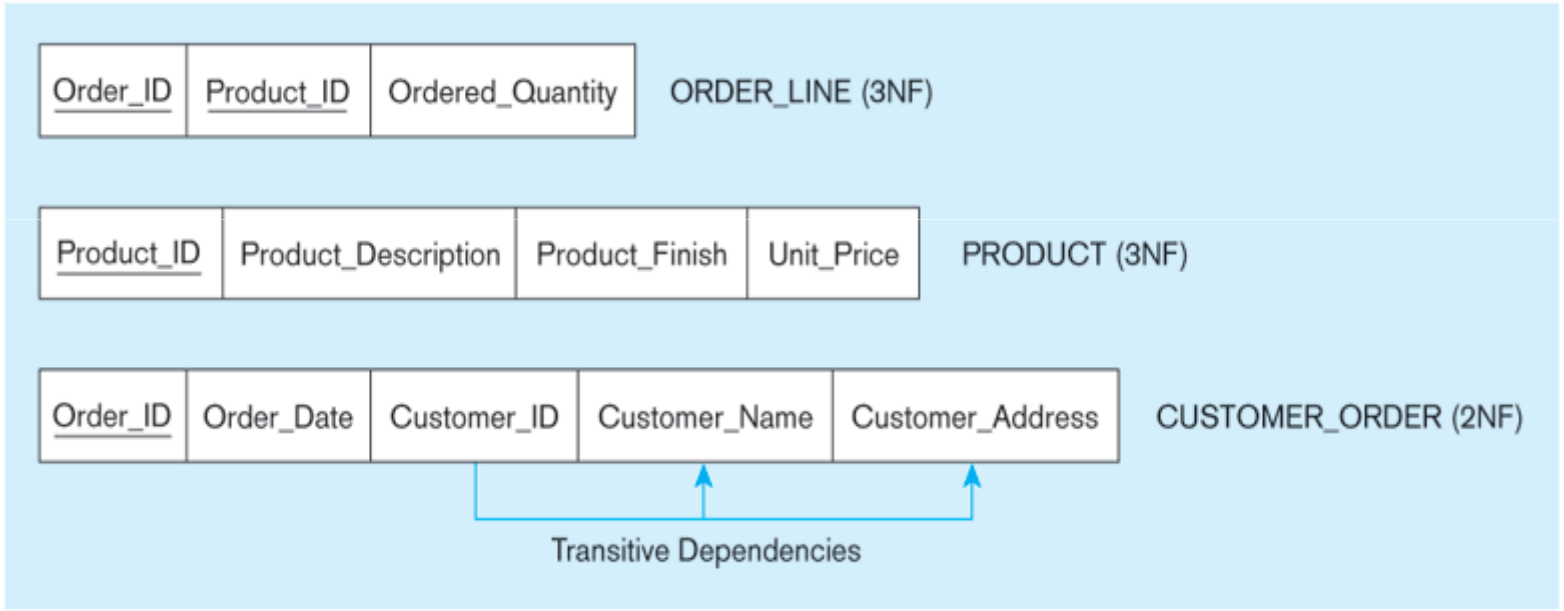
Product_ID ➔ Product_Description, Product_Finish, Unit_Price

Order_ID, Product_ID ➔ Order_Quantity

Therefore, NOT in 2nd Normal Form

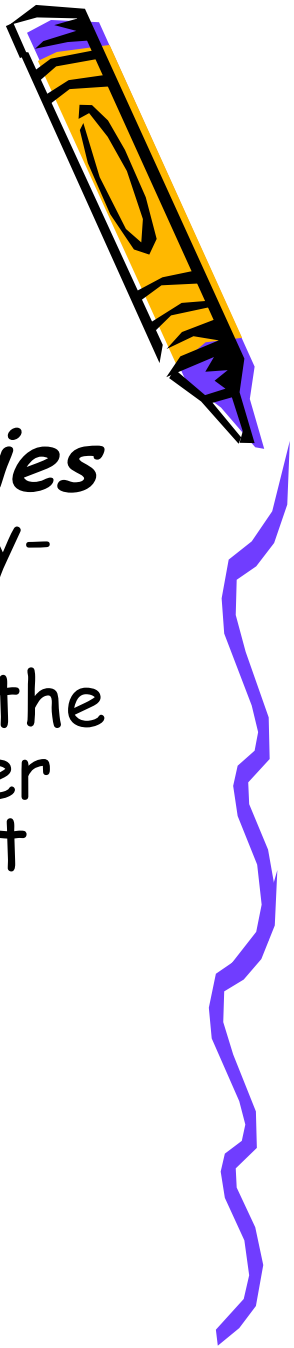
Getting it into Second Normal Form

Partial Dependencies must be removed

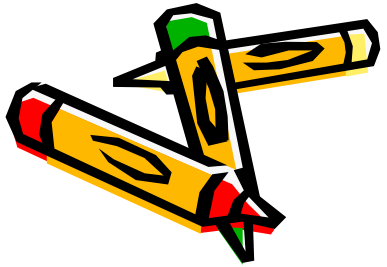


but there are still transitive dependencies

Third Normal Form

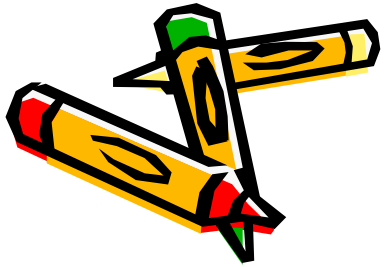


- 2NF PLUS ***no transitive dependencies*** (functional dependencies on non-primary-key attributes)
- Note: this is called transitive, because the primary key is a determinant for another attribute, which in turn is a determinant for a third

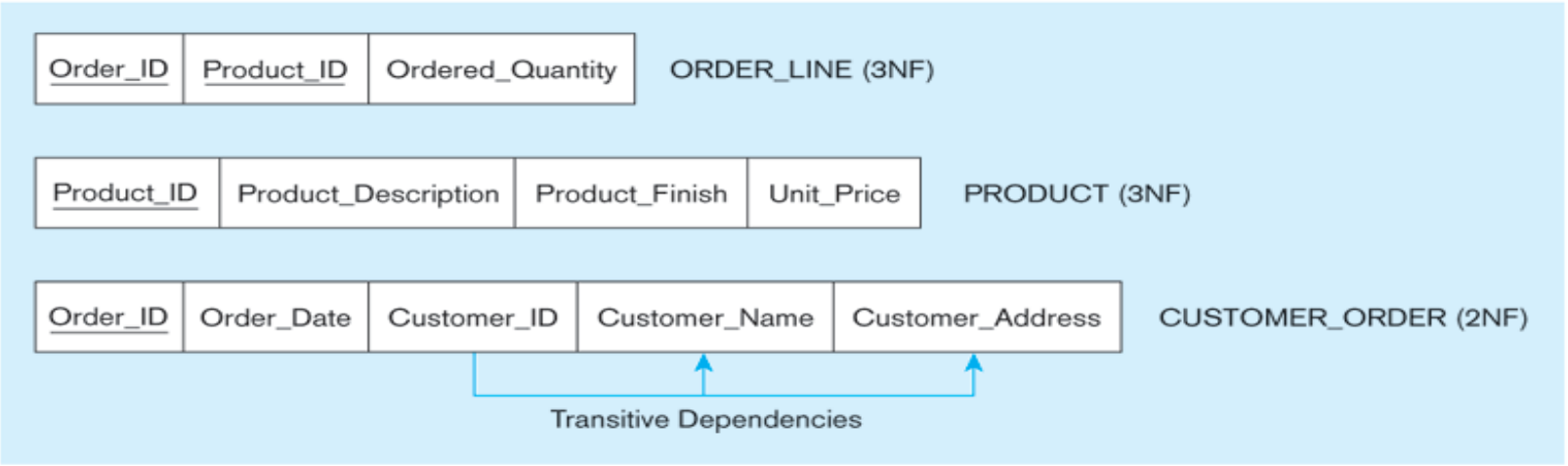


Third Normal Form

- Solution: non-key determinant with transitive dependencies go into a new table; non-key determinant becomes primary key in the new table and stays as foreign key in the old table



Getting it into Third Normal Form



Transitive dependencies are removed