**Chapter 4 Exercises Page - 157**

1. SELECT \*

FROM Vendors JOIN Invoices

ON Vendors.VendorID = Invoices.VendorID

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. SELECT Vendors.VendorName,

Invoices.InvoiceNumber,

Invoices.InvoiceDate,

Invoices.InvoiceTotal - ( Invoices.PaymentTotal + Invoices.CreditTotal )

AS Balance

FROM Vendors JOIN Invoices

ON Vendors.VendorID = Invoices.VendorID

WHERE Invoices.InvoiceTotal - (Invoices.PaymentTotal + Invoices.CreditTotal) > 0

ORDER BY Vendors.VendorName

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. Select Vendors.VendorName,

Vendors.DefaultAccountNo,

GLAccounts.AccountDescription

FROM Vendors JOIN GLAccounts

ON Vendors.DefaultAccountNo = GLAccounts.AccountNo

ORDER BY GLAccounts.AccountDescription,

Vendors.VendorName

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. SELECT Vendors.VendorName,

Invoices.InvoiceNumber,

Invoices.InvoiceDate,

Invoices.InvoiceTotal - ( Invoices.PaymentTotal + Invoices.CreditTotal )

AS Balance

FROM Vendors, Invoices

WHERE Vendors.VendorID = Invoices.VendorID AND

Invoices.InvoiceTotal - (Invoices.PaymentTotal + Invoices.CreditTotal) > 0

ORDER BY Vendors.VendorName

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. SELECT v.VendorName AS Vendor,

i.InvoiceDate AS Date,

i.InvoiceNumber AS Number,

il.InvoiceSequence AS #,

il.InvoiceLineItemAmount AS LineItem

FROM Vendors AS v

JOIN Invoices AS i

ON v.VendorID = i.VendorID

JOIN InvoiceLineItems AS il

ON i.InvoiceID = il.InvoiceID

ORDER BY Vendor,

Date,

Number,

#

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. SELECT DISTINCT v1.VendorID,

v1.VendorName,

v1.VendorContactFName + ' ' + v1.VendorContactLName AS Name

FROM Vendors AS v1 JOIN Vendors AS v2 ON

( v1.VendorContactFName = v2.VendorContactFName ) AND

( v1.VendorID <> v2.VendorID )

ORDER BY Name

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. SELECT GLAccounts.AccountNo,

GLAccounts.AccountDescription

FROM GLAccounts LEFT JOIN InvoiceLineItems ON

GLAccounts.AccountNo = InvoiceLineItems.AccountNo

WHERE InvoiceLineItems.AccountNo IS NULL

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. SELECT Vendors.VendorName,

'CA' AS VendorState

FROM Vendors

WHERE Vendors.VendorState = 'CA'

UNION

SELECT Vendors.VendorName,

'Outside CA' AS VendorState

FROM Vendors

WHERE Vendors.VendorState <> 'CA'

ORDER BY Vendors.VendorName

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Chapter 5**

**Vocabulary matching**

1 - E

2 - C

3 - A

4 - I

5 - J

6 - H

7 - G

8 - F

9 - D

10 - B

**Practices**

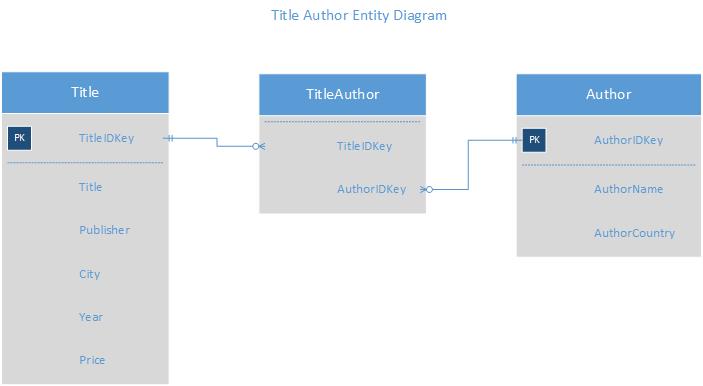
1. Violates all normal forms and there are multiple values in the “Titles” colomn.
2. Titles colomn is maultivalued

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Title ID Key | Title | Publisher | City | Year | Price |
| 1 | JavaScript Essentials | South Tech Books | London | 2010 | 14 |
| 2 | HTML5 Exposed | Webby Books | London | 2012 | 15.5 |
| 3 | BigData Big Promise | Data Press | SFO | 2012 | 25 |
| 4 | Database development for the cloud | Data Press | SFO |  | 20.35 |
| 5 | Data Services | Future Tech Press | New York |  | 12.95 |

|  |  |  |
| --- | --- | --- |
| AuthorID Key | AuthorName | Author Country |
| 1 | JamesTaylor | England |
| 2 | May Norton | United States |
| 3 | Jessica Lewis | United States |

|  |  |
| --- | --- |
| Title ID Key | Author ID Key |
| 1 | 1 |
| 2 | 1 |
| 3 | 2 |
| 4 | 3 |
| 5 | 3 |

1. Entity Diagram for the above table

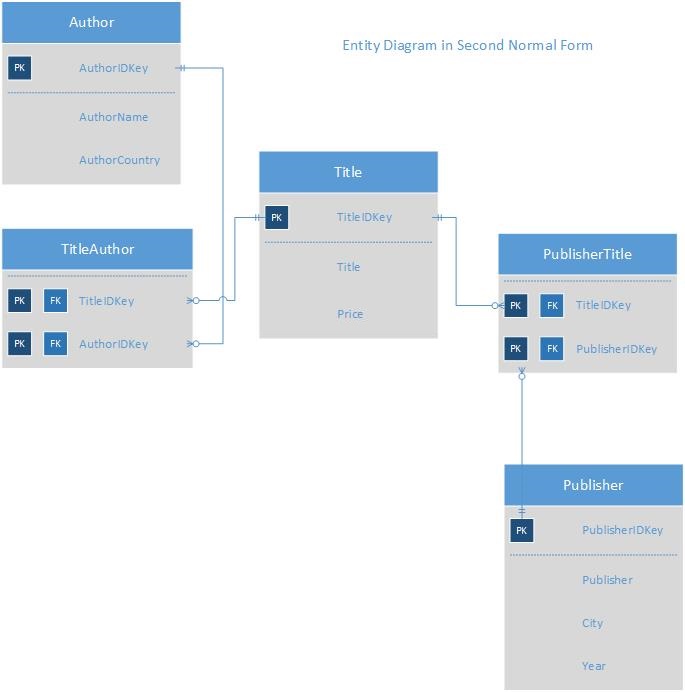


* Author’s name can be uniquely determined from the book and not the other way around
* Publisher’s name can be uniquely determined from the book and not the other way around

1. Candidate Keys

* TilteIDKey
* AuthorIDKey

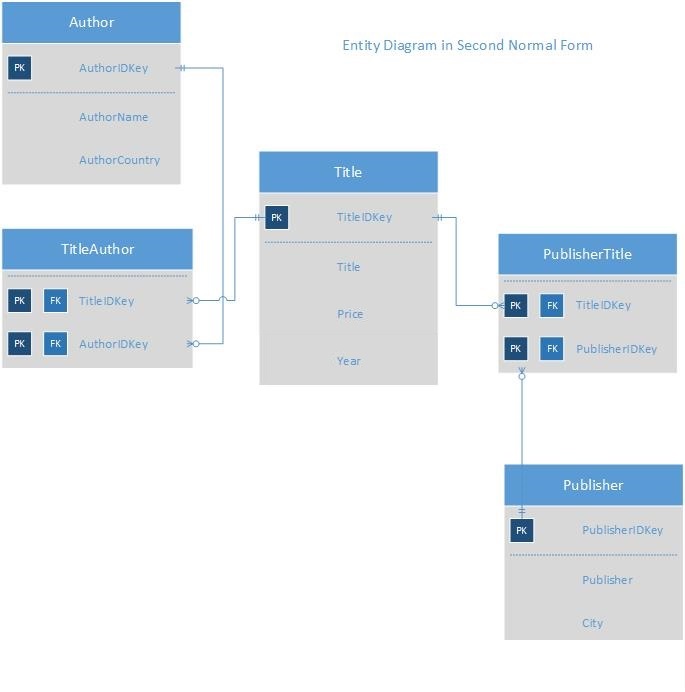
1. Entity diagram in second normal form



1. Transitive dependency

The “year” attribute in the publisher table is not directly related to the ‘publisher’ but instead to the ‘title’ table which is a transitive dependency.

1. Entity diagram in Third Normal Form



1. Processes I went through to achieve the third normal form

* Designed the table in such a way to eliminate multiple values from single colomn
* Drew the entity diagram from that table
* Listed the functional deficiency
* Drew the entity diagram to address the functional deficiency which resulted in the second normal form
* Listed the transitive deficiency
* Drew the entity diagram to address the transitive deficiency which resulted in the third normal form

**Chapter 5**

**Vince’s Vinyl**

Review and change the vince’s vinyl entity diagram to reflect 3rd normalised form and document the changes.

I split the Vinyl Record table further to Artist table and Record(Album) table due to the possible nature of a vinyl record having multiple albums in a single vinyl record and also not to rule out the possibility of multiple artists for a single album or record.

(Note to Instructor :There are a few discrepancies I could still think of in terms of naming the entities which I could have done at the first place while designing the Vince’s entity diagram. Maybe a customer in place of buyer, product in place of vinyl record, and including address as a separate table. Those would have helped better structure and understand the table and to normalise it in a better way. I believe, one would get better with experience. Thanks for the exposure)

Again to my knowledge, I believe the below entity diagram meets the standards of a third normal form and should fulfill the needs of vince’s database requirements.

**Vince’s Vinyl Entity diagram in Third Normal Form**

