Modeling an Online Music Business

Zane Kansil
Loyola Marymount University
Database Systems

December 15, 2014

Contents

I Title Page **1**

II Contents **2**

III Description of the Enterprise **3**
III.1 Ten sample Questions John would ask 4

IV Definition of Environment **5**
IV.1 Input and Report forms 5
IV.2 Assumptions 7
IV.3 User-oriented data dictionary 8
IV.4 Cross-reference table 10

V Enterprise Database Design **12**
V.1 Logical model of the Enterprise 12
V.1.1 List of Entities and Attributes 12
V.1.2 List of Relationships and Attributes 14
V.1.3 Entity-Relationship diagram of the Enterprise 15
V.2 Conceptual model of the enterprise 16
V.3 Table dictionary 18
V.4 Attribute dictionary 19

VI Database and Query Definition **21**
VI.1 Database Definition 21
VI.2 Database Queries 28
VI.3 Design Tradeoffs and Limitations 31

VII Database Integrity and Security **32**
VII.1 Functional Dependencies 32
VII.2 Adjustments for Normalization 33
VII.3 Integrity and Security 34

VIII Implementation Notes **35**
VIII.1 Indices 35
VIII.2 Data 36
VIII.3 Query Trace 40
VIII.4 Implementation Assessment 41

IX Lessons Learned **42**

Chapter III

Description of the Enterprise

A friend of mine intends to put out an all-purpose site for his musical career. The site will be a hub for his online business and allow him to track sales and interactions with his fans.

The site will serve as a place to feature embedded music and video players for John's music. It will be necessary to monitor this hosted media in terms of listens and views. Some other metrics John is interested in are on-site plays (not a redirect), and redirects to his Soundcloud and YouTube accounts from their embedded players on his site. Videos and Songs are tracked separately due to their differing properties.

There is a sales aspect to the site, merchandise will also be sold under the same domain. Some merchandise will be physical, such as hats, headbands, wristbands and stickers. Other merchandise will be digital, such as a donation to download a mixtape (a non-album collection of John's music). Physical merchandise needs to be shipped, so appropriate data such as `shipping_status` and `destination` will need to be tracked. With physical merchandise, quantity and availability must be tracked. Digital merchandise is easier to manage as it is transmitted and there is less customer data to collect. Digital merchandise will also be infinitely available if listed, so there is no quantity related data to track.

JohnDB records data for Physical and Digital consumers separately.

As the main operator of his business, John wants to track of his sales and revenue. The most simple way to track this would be through a table of sales records. These records would detail everything necessary about the sale. A sale would correspond to a single type of product. If multiple products were purchased in the same transaction, then multiple sale entries would be logged.

III.1 Ten sample Questions John would ask

1. How many video plays today?
2. How many redirects to Soundcloud from embedded music players?
3. What is the redirect rate for videos?
4. Is “Black Silk Hooded Sweatshirt” sold out?
5. How many “Red Summer Beanie” items were sold in October 2014?
6. What digital goods are currently unavailable?
7. How many orders do I have to fill to the US?
8. What precentage of my digital consumers are from outside the US?
9. How many digital consumers are also physical consumers?
10. What products have garnered zero sales in the past 14 days?

Chapter IV

Definition of Environment

IV.1 Input and Report forms

1. Video Metadata View

    - Video Name
    - Video (hosted at) URL
    - Number of plays
    - Total plays
    - Plays Today
    - Total redirects to YouTube

2. Song Metadata View

    - Song Name
    - Song Artist
    - Song (hosted at) URL
    - Total plays
    - Plays Today
    - Total redirects to SoundCloud

3. Add New Physical good

    - Name
    - Description Paragraph
    - Color (optional)
    - Size (optional)
    - Price (in USD)
    - Current stock on hand

4. Add New Digital good

    - Name
    - Description Paragraph
    - Price (in USD)

5. Physical Good Admin View

    - Name
    - Good SKU

- Good description (editable)
- Price (in USD)
- Color of Good (editable)
- Quantity
- Size of Good

6. Digital Good Admin View

    - Name
    - Good SKU
    - Good description (editable)
    - Price (in USD)
    - Available (togglable)

7. Physical Consumer Admin View

    - Customer id
    - First Name (editable)
    - Last Name (editable)
    - Customer Phone Number (editable)
    - Address Line 1 (editable)
    - Address Line 2 (editable)
    - Customer Country (fixed to 'US')
    - Customer State (US states including 'HI' and 'AL')
    - Customer Zip Code (editable)

8. Digital Consumer Admin View

    - Customer id
    - First Name (editable)
    - Last Name (editable)
    - Customer Phone Number (editable)
    - Customer Country (editable)

9. Transaction Log

    - Item SKU
    - Sale type (`digital` or `physical`, derived)
    - Status (`received`, `shipped` or `fulfilled`)
    - Availability of Good
    - Quantity (of each line item)
    - Unit Price (of each line item)
    - Line number (each line item is numbered sequentially)
    - Total Order Cost (derived)
    - Sale date (time that purchase was made)

10. Interaction Log

    - Interaction timestamp
    - Media identifier (interaction was made against this Song or Video)
    - Title of Media

IV.2 Assumptions

1. Forms are used to add items to the site
2. Tables as opposed to graphs are the prefered way to view data
3. The Sales table functions as an Orders table as well, showing the status of each order in addition to transaction information
4. Only customers in the US are allowed to order physical goods

IV.3 User-oriented data dictionary

Datum	**Information Definition**
c_email	Email address
c_first_name	First name of customer
c_id	Identifies a customer
c_last_name	Last name of customer
c_phone	Phone number. All digits, no dashes or spaces.
dc_country	Country which customer resides. Optional and may be any country.
dc_id	Alias of 'c_id', used specifically to identify a digital consumer
dg_id	Alias of 'g_id', used specifically to identify a digital good
dg_is_available	Reflects whether or not this good is available for purchase. Set to "false" to prevent customers from soliciting a copy
g_description	Description of good
g_name	Name of physical good
g_price	Price of good in USD pennies
g_sku	Uniquely identifies a class of item for sale. For physical goods this is specific for each color color size.
i_date	Datetime this interaction was logged
i_id	Interaction identifier.
li_number	Identifies the particular line item amongst a list of line items. Line items in a purchase are numbered sequentially in this manner
li_quantity	Number of units sold in the line item. A series of line items composes a purchase.
m_id	Media identifier. Identifies a Song or Video hosted on the site.
m_upload_date	Date/time that the video was uploaded
mv_plays	Total plays originating at the site. This is derived from the log of PLAY entries pointing to this song.
mv_plays_today	Plays originating at the site today. This is derived from the log of PLAY entries pointing to this song.
mv_redirects	Number of redirects to YouTube (which occurs when the embedded video is clicked by a viewer). This is derived from the log of REDIRECT entries pointing to this song.
mv_title	Music videos title, identical to its title on Youtube
mv_url	YouTube URL that the video is hosted at
pc_address_line_1	Address of customer
pc_address_line_2	Second line of customer address
pc_country	Country which customer resides. This will always be "US"
pc_id	Alias of 'c_id', used specifically to identify a physical consumer
pc_state	State which customer resides
pc_zip_code	Zip code. 5 digits in the US.
pg_color	Color of good
pg_id	Alias of 'g_id', used specifically to identify a physical good

pg_quantity_available	Current quantity available. This is an editable field so users should take care not to set the field inappropriately.
pg_size	Size of good, either 'S', 'M','L', ..., or a numbered size, or 'OSFA' (one size fits all)
pl_id	Alias of 'i_id', used specifically to identify a play
re_id	Alias of 'i_id', used specifically to identify a redirect
re_url	The non-local url that the redirect sent the user to. Music videos redirect to a Soundcloud uri, Videos redirect to a Youtube uri.
sale_date	Datetime this sale was logged
sale_fulfill_date	Date this sale was set to 'fulfilled'
sale_id	Sales/transaction identifier
sale_status	The shipping status, 'received', 'shipped' or 'fulfilled'
so_artist	Artist of song, including features
so_plays	Total plays originating at the site. This is derived from the log of PLAY entries pointing to this song.
so_plays_today	Plays on the site today. This is derived from the log of PLAY entries pointing to this song.
so_redirects	Redirects to SoundCloud (triggered by clicks on the embedded player). This is derived from the log of REDIRECT entries pointing to this song.
so_title	Title of song
so_url	SoundCloud URL that the video is hosted at

IV.4 Cross-reference table

Datum	Form or Screen									
	Video Metadata View	Song Metadata View	Add New Physical good	Add New Digital good	Physical Good Admin	Digital Good Admin	Physical Consumer Admin View	Digital Consumer Admin View	Transaction Log	Interaction Log
c_email							x	x		
c_first_name							x	x		
c_id										
c_last_name							x	x		
c_phone							x	x		
dc_country								x		
dc_id								x		
dg_id						x				
dg_is_available				x		x			x	
g_description			x	x	x	x				
g_name			x	x	x	x			x	
g_price			x	x	x	x			x	
g_sku									x	
i_date										x
i_id										x
li_quantity									x	
m_id										x
m_upload_date		x								
mv_plays	x									
mv_plays_today	x									
mv_redirects	x									
mv_title	x									x
mv_url	x									
pc_address_line_1							x			
pc_address_line_2							x			
pc_country							x			
pc_id							x			
pc_state							x			
pc_zip_code							x			
pg_color			x		x				x	
pg_id					x					
pg_quantity_available			x		x					

pg_size			x		x				x	
re_url										
sale_date									x	
sale_fulfill_date									x	
sale_id									x	
sale_status									x	
so_artist		x								
so_plays		x								
so_plays_today		x								
so_redirects		x								
so_title		x								x
so_url		x								

Chapter V

Enterprise Database Design

V.1 Logical model of the Enterprise

V.1.1 List of Entities and Attributes

1. Media
    - m_id
2. Music Video
    - mv_id (alias of m_id)
    - mv_title
    - mv_url (url video is hosted at)
    - mv_upload_date
3. Song
    - so_id (alias of m_id)
    - so_title
    - so_artist
    - so_url (url song is hosted at)
    - so_upload_date
4. Interaction
    - i_id
    - i_date
5. Play
    - pl_id (alias of i_id)
6. Redirect
    - re_id (alias of i_id)
    - re_url
7. Consumer
    - c_id
    - c_firstname

- c_lastname
- c_email
- c_phone

8. Physical Consumer

    - pc_id (alias of c_id)
    - pc_address_line_1
    - pc_address_line_2
    - pc_country
    - pc_state

9. Digital Consumer

    - dc_id (alias of c_id)
    - dc_country (optional)

10. Purchase

    - sale_id
    - sale_status ("received", "shipped" or "fulfilled")
    - sale_date (datetime)
    - sale_fulfill_date (datetime)

11. Line Item

    - li_quantity

12. Good

    - g_sku
    - g_name
    - g_description
    - g_price

13. Digital Good

    - dg_id (alias g_sku)
    - dg_is_available (boolean, used to prevent ordering)

14. Physical Good

    - pg_id (alias g_sku)
    - pg_color
    - pg_size
    - pg_quantity_available

* - A datetime is an instant in time. Has date information and time-of-day information. Example: 2014-09-06T15:35:58+00:00 (September 6, 2014, 3:35:58pm)

V.1.2 List of Relationships and Attributes

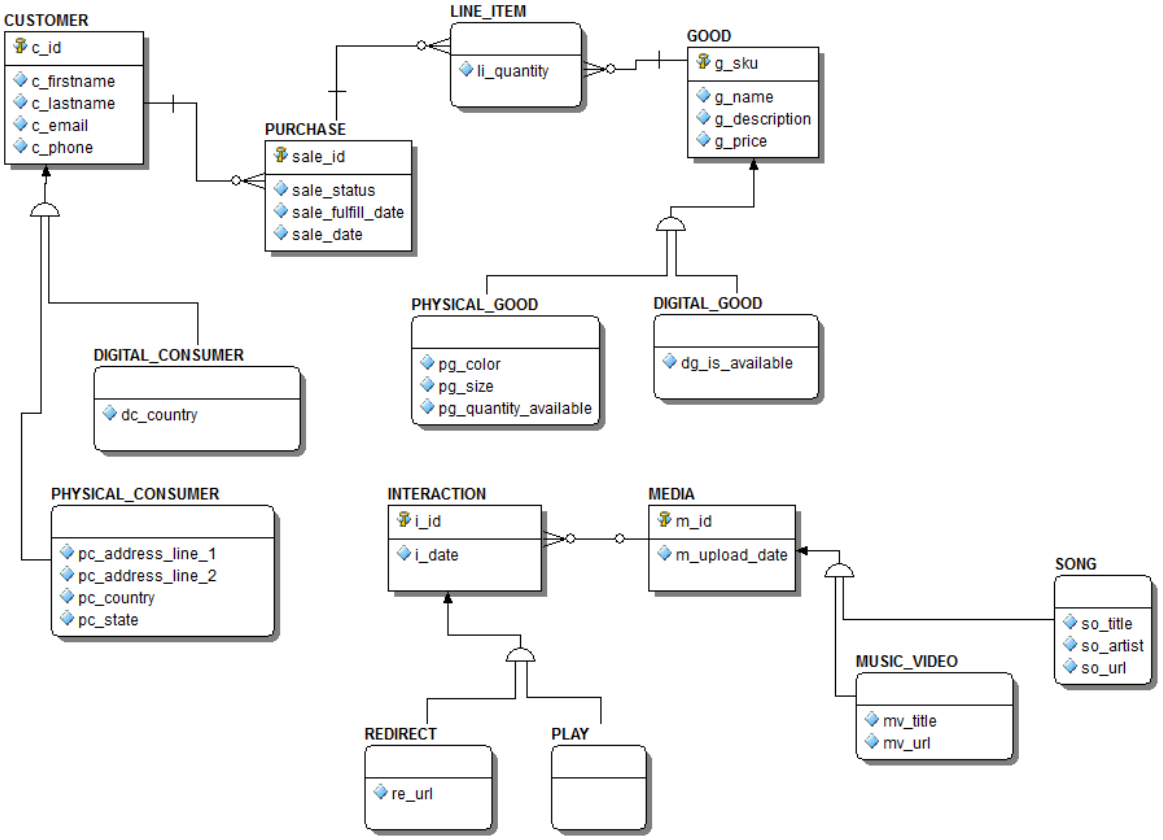
Media Relationships

1. INTERACTION(i_id, m_id)

Sales Relationships

1. CUSTOMER_PURCHASE(c_id, sale_id)
2. LINE_ITEM(sale_id, g_sku)

V.1.3 Entity-Relationship diagram of the Enterprise



V.2 Conceptual model of the enterprise

```
MEDIA(m_id, m_upload_date)

MUSIC_VIDEO(mv_id, mv_title, mv_url)
    PK/FK: mv_id
    CK:    mv_id, mv_title, mv_url

SONG(so_id, so_title, so_artist, so_url)
    PK/FK: so_id
    CK:    so_id, so_title, so_url

PLAY(pl_id, m_id, i_datetime)
    PK: pl_id
    FK: m_id
    CK: pl_id

REDIRECT(m_id, m_to)
    PK: re_id
    FK: m_id
    CK: re_id

CONSUMER(c_id, c_firstname, c_lastname, c_email)
    PK: c_id
    CK: c_id, c_email

PHYSICAL_CONSUMER(
  pc_id
, pc_address_line_1
, pc_address_line_2
, pc_country
, pc_state
, pc_phone
)
    PK/FK: pc_id
    CK: pc_id, pc_phone

DIGITAL_CONSUMER(
  dc_id
, pc_phone
, pc_country
)
    PK/FK: dc_id
    CK:    dc_id, pc_phone

PURCHASE(sale_id, c_id, sale_date, sale_fulfill_date)
    PK/FK: sale_id
    CK:    sale_id, c_id
```

```
LINE_ITEM(sale_id, g_id, li_quantity)
    PK: sale_id
    CK: sale_id
    FK: sale_id, g_id

GOOD(g_sku, g_name, g_description, g_price)
    PK: g_sku
    CK: g_sku

DIGITAL_GOOD(dg_id, dg_is_available)
    PK: dg_id
    CK: dg_id

PHYSICAL_GOOD(pg_id, pg_color, pg_size, pg_quantity_available)
    PK/FK: pg_id
    CK:    pg_id
```

V.3 Table dictionary

Table	Attributes	Description
CUSTOMER	c_id, c_firstname, c_lastname, c_email, c_phone	Basic customer information
DIGITAL_CONSUMER	dc_id, dc_country	Customer information specific to digital customers
DIGITAL_GOOD	dg_id, dg_is_available	Good information specific to digital goods
GOOD	g_sku, g_name, g_description, g_price	Basic information for goods that are being sold
INTERACTION	i_id, m_id, i_date	A list of all of the interactions made on media items
LINE_ITEM	sale_id, g_sku, li_quantity	An item that was sold in a transaction
MEDIA	m_id, m_upload_date	Basic information for songs and videos on the page
MUSIC_VIDEO	mv_id, mv_title, mv_url	Additional media information specific to Music Videos
PHYSICAL_CONSUMER	pc_id, pc_address_line_1, pc_address_line_2, pc_country, pc_state	Customer information specific to physical customers
PHYSICAL_GOOD	pg_id, pg_color, pg_size, pg_quantity_available	Good information specific to physical goods
PLAY	pl_id	A table of all the play interactions
PURCHASE	sale_id, sale_status, sale_fulfill_date, sale_date, c_id	Information specific to a single sale
REDIRECT	re_id, re_url	A table of all the redirect interactions
SONG	so_id, so_title, so_artist, so_url	Additional media information specific to Songs

V.4 Attribute dictionary

Datum	**Appears in**	**Information Definition**
c_email	CUSTOMER	Email address
c_first_name	CUSTOMER	First name of customer
c_id	CUSTOMER, PURCHASE	Identifies a customer
c_last_name	CUSTOMER	Last name of customer
c_phone	CUSTOMER	Phone number. All digits, no dashes or spaces.
dc_country	DIGITAL_CUSTOMER	Country which customer resides. Optional and may be any country.
dc_id	DIGITAL_CUSTOMER	Alias of 'c_id', used specifically to identify a digital consumer
dg_id	DIGITAL_GOOD	Alias of 'g_id', used specifically to identify a digital good
dg_is_available	DIGITAL_GOOD	Reflects whether or not this good is available for purchase. Set to "false" to prevent customers from soliciting a copy
g_description	GOOD	Description of good
g_name	GOOD	Name of physical good
g_price	GOOD	Price of good in USD pennies
g_sku	GOOD, LINE_ITEM	Uniquely identifies a class of item for sale. For physical goods this is specific for each color color size.
i_date	INTERACTION	Datetime this interaction was logged
i_id	INTERACTION	Interaction identifier.
li_quantity	LINE_ITEM	Number of units sold in the line item. A series of line items composes a purchase.
m_id	MEDIA, INTERACTION	Media identifier. Identifies a Song or Video hosted on the site.
m_upload_date	MEDIA	Date/time that the video was uploaded
mv_plays	MUSIC_VIDEO	Total plays originating at the site. This is derived from the log of PLAY entries pointing to this song.
mv_plays_today	MUSIC_VIDEO	Plays originating at the site today. This is derived from the log of PLAY entries pointing to this song.
mv_redirects	MUSIC_VIDEO	Number of redirects to YouTube (which occurs when the embedded video is clicked by a viewer). This is derived from the log of REDIRECT entries pointing to this song.
mv_title	MUSIC_VIDEO	Music videos title, identical to its title on Youtube
mv_url	MUSIC_VIDEO	YouTube URL that the video is hosted at
pc_address_line_1	PHYSICAL_CONSUMER	Address of customer
pc_address_line_2	PHYSICAL_CONSUMER	Second line of customer address

pc_country	PHYSICAL_CONSUMER	Country which customer resides. This will always be "US"
pc_id	PHYSICAL_CONSUMER	Alias of 'c_id', used specifically to identify a physical consumer
pc_state	PHYSICAL_CONSUMER	State which customer resides
pc_zip_code	PHYSICAL_CONSUMER	Zip code. 5 digits in the US.
pg_color	PHYSICAL_GOOD	Color of good
pg_id	PHYSICAL_GOOD	Alias of 'g_id', used specifically to identify a physical good
pg_quantity_available	PHYSICAL_GOOD	Current quantity available. This is an editable field so users should take care not to set the field inappropriately.
pg_size	PHYSICAL_GOOD	Size of good, either 'S', 'M','L', ..., or a numbered size, or 'OSFA' (one size fits all)
pl_id	PLAY	Alias of 'i_id', used specifically to identify a play
re_id	REDIRECT	Alias of 'i_id', used specifically to identify a redirect
re_url	REDIRECT	The non-local url that the redirect sent the user to. Music videos redirect to a Soundcloud uri, Videos redirect to a Youtube uri.
sale_date	PURCHASE	Datetime this sale was logged
sale_fulfill_date	PURCHASE	Date this sale was set to 'fulfilled'
sale_id	PURCHASE, LINE_ITEM	Sales/transaction identifier
sale_status	PURCHASE	The shipping status, 'received', 'shipped' or 'fulfilled'
so_artist	SONG	Artist of song, including features
so_plays	SONG	Total plays originating at the site. This is derived from the log of PLAY entries pointing to this song.
so_plays_today	SONG	Plays on the site today. This is derived from the log of PLAY entries pointing to this song.
so_redirects	SONG	Redirects to SoundCloud (triggered by clicks on the embedded player). This is derived from the log of REDIRECT entries pointing to this song.
so_title	SONG	Title of song
so_url	SONG	SoundCloud URL that the video is hosted at

Chapter VI

Database and Query Definition

VI.1 Database Definition

```
--
-- ER/Studio Data Architect 9.6 SQL Code Generation
-- Project :      zk-online-music-business.DM1
--
-- Date Created : Tuesday, December 09, 2014 16:44:51
-- Target DBMS : MySQL 5.x
--

--
-- TABLE: CUSTOMER
--

CREATE TABLE CUSTOMER(
    c_id           CHAR(20)       NOT NULL,
    c_firstname    VARCHAR(20)    NOT NULL,
    c_lastname     VARCHAR(20)    NOT NULL,
    c_email        VARCHAR(20)    NOT NULL,
    c_phone        VARCHAR(20)    NOT NULL,
    PRIMARY KEY (c_id)
)ENGINE=INNODB
;



--
-- TABLE: DIGITAL_CONSUMER
--

CREATE TABLE DIGITAL_CONSUMER(
    dc_id         CHAR(20)       NOT NULL,
    dc_country    VARCHAR(20)    NOT NULL,
    PRIMARY KEY (dc_id)
)ENGINE=INNODB
;
```

```
--
-- TABLE: DIGITAL_GOOD
--

CREATE TABLE DIGITAL_GOOD(
    dg_id              CHAR(20)    NOT NULL,
    dg_is_available    BIT(1)      NOT NULL,
    PRIMARY KEY (dg_id)
)ENGINE=INNODB
;



--
-- TABLE: GOOD
--

CREATE TABLE GOOD(
    g_sku            CHAR(20)         NOT NULL,
    g_name           VARCHAR(255)     NOT NULL,
    g_description    VARCHAR(2048)    NOT NULL,
    g_price          INT              NOT NULL,
    PRIMARY KEY (g_sku)
)ENGINE=INNODB
;



--
-- TABLE: INTERACTION
--

CREATE TABLE INTERACTION(
    i_id      CHAR(20)    NOT NULL,
    m_id      CHAR(20),
    i_date    DATETIME    NOT NULL,
    PRIMARY KEY (i_id)
)ENGINE=INNODB
;



--
-- TABLE: LINE_ITEM
--
```

```
CREATE TABLE LINE_ITEM(
    sale_id        CHAR(20)    NOT NULL,
    g_sku          CHAR(20)    NOT NULL,
    li_quantity    INT,
    PRIMARY KEY (sale_id, g_sku)
)ENGINE=INNODB
;



--
-- TABLE: MEDIA
--

CREATE TABLE MEDIA(
    m_id             CHAR(20)    NOT NULL,
    m_upload_date    DATETIME,
    PRIMARY KEY (m_id)
)ENGINE=INNODB
;



--
-- TABLE: MUSIC_VIDEO
--

CREATE TABLE MUSIC_VIDEO(
    mv_id       CHAR(20)        NOT NULL,
    mv_title    VARCHAR(50)     NOT NULL,
    mv_url      VARCHAR(100)    NOT NULL,
    PRIMARY KEY (mv_id)
)ENGINE=INNODB
;



--
-- TABLE: PHYSICAL_CONSUMER
--

CREATE TABLE PHYSICAL_CONSUMER(
    pc_id                CHAR(20)        NOT NULL,
    pc_address_line_1    VARCHAR(255)    NOT NULL,
    pc_address_line_2    VARCHAR(255),
    pc_country           VARCHAR(20)     NOT NULL,
    pc_state             VARCHAR(20)     NOT NULL,
    PRIMARY KEY (pc_id)
)ENGINE=INNODB
```

```
;



--
-- TABLE: PHYSICAL_GOOD
--

CREATE TABLE PHYSICAL_GOOD(
    pg_id                   CHAR(20)       NOT NULL,
    pg_color                VARCHAR(20),
    pg_size                 VARCHAR(20),
    pg_quantity_available   INT            NOT NULL,
    PRIMARY KEY (pg_id)
)ENGINE=INNODB
;



--
-- TABLE: PLAY
--

CREATE TABLE PLAY(
    pl_id    CHAR(20)    NOT NULL,
    PRIMARY KEY (pl_id)
)ENGINE=INNODB
;



--
-- TABLE: PURCHASE
--

CREATE TABLE PURCHASE(
    sale_id              CHAR(20)       NOT NULL,
    sale_status          VARCHAR(20)    NOT NULL,
    sale_fulfill_date    DATETIME,
    sale_date            DATETIME       NOT NULL,
    c_id                 CHAR(20)       NOT NULL,
    PRIMARY KEY (sale_id)
)ENGINE=INNODB
;



--
-- TABLE: REDIRECT
```

```
--

CREATE TABLE REDIRECT(
    re_id     CHAR(20)        NOT NULL,
    re_url    VARCHAR(255)    NOT NULL,
    PRIMARY KEY (re_id)
)ENGINE=INNODB
;



--
-- TABLE: SONG
--

CREATE TABLE SONG(
    so_id        CHAR(20)        NOT NULL,
    so_title     VARCHAR(50)     NOT NULL,
    so_artist    VARCHAR(20)     NOT NULL,
    so_url       VARCHAR(100)    NOT NULL,
    PRIMARY KEY (so_id)
)ENGINE=INNODB
;



--
-- TABLE: DIGITAL_CONSUMER
--

ALTER TABLE DIGITAL_CONSUMER ADD CONSTRAINT RefCUSTOMER2
    FOREIGN KEY (dc_id)
    REFERENCES CUSTOMER(c_id)
;


--
-- TABLE: DIGITAL_GOOD
--

ALTER TABLE DIGITAL_GOOD ADD CONSTRAINT RefGOOD7
    FOREIGN KEY (dg_id)
    REFERENCES GOOD(g_sku)
;


--
-- TABLE: INTERACTION
--
```

```
ALTER TABLE INTERACTION ADD CONSTRAINT RefMEDIA15
    FOREIGN KEY (m_id)
    REFERENCES MEDIA(m_id)
;


--
-- TABLE: LINE_ITEM
--

ALTER TABLE LINE_ITEM ADD CONSTRAINT RefPURCHASE4
    FOREIGN KEY (sale_id)
    REFERENCES PURCHASE(sale_id)
;

ALTER TABLE LINE_ITEM ADD CONSTRAINT RefGOOD5
    FOREIGN KEY (g_sku)
    REFERENCES GOOD(g_sku)
;


--
-- TABLE: MUSIC_VIDEO
--

ALTER TABLE MUSIC_VIDEO ADD CONSTRAINT RefMEDIA10
    FOREIGN KEY (mv_id)
    REFERENCES MEDIA(m_id)
;


--
-- TABLE: PHYSICAL_CONSUMER
--

ALTER TABLE PHYSICAL_CONSUMER ADD CONSTRAINT RefCUSTOMER1
    FOREIGN KEY (pc_id)
    REFERENCES CUSTOMER(c_id)
;


--
-- TABLE: PHYSICAL_GOOD
--

ALTER TABLE PHYSICAL_GOOD ADD CONSTRAINT RefGOOD6
    FOREIGN KEY (pg_id)
    REFERENCES GOOD(g_sku)
```

```
;


--
-- TABLE: PLAY
--

ALTER TABLE PLAY ADD CONSTRAINT RefINTERACTION12
    FOREIGN KEY (pl_id)
    REFERENCES INTERACTION(i_id)
;


--
-- TABLE: PURCHASE
--

ALTER TABLE PURCHASE ADD CONSTRAINT RefCUSTOMER3
    FOREIGN KEY (c_id)
    REFERENCES CUSTOMER(c_id)
;


--
-- TABLE: REDIRECT
--

ALTER TABLE REDIRECT ADD CONSTRAINT RefINTERACTION13
    FOREIGN KEY (re_id)
    REFERENCES INTERACTION(i_id)
;


--
-- TABLE: SONG
--

ALTER TABLE SONG ADD CONSTRAINT RefMEDIA11
    FOREIGN KEY (so_id)
    REFERENCES MEDIA(m_id)
;
```

VI.2 Database Queries

```
-- (1) How many video plays today?

SELECT     COUNT(‘i‘.‘i_date‘) AS ‘plays‘
FROM       ‘MUSIC_VIDEO‘ AS ‘mv‘
INNER JOIN ‘INTERACTION‘ AS ‘i‘
ON         ‘mv‘.‘mv_id‘  =  ‘i‘.‘m_id‘
INNER JOIN ‘PLAY‘        AS ‘pl‘
ON         ‘pl‘.‘pl_id‘  =  ‘i‘.‘i_id‘
WHERE      DATE(‘i‘.‘i_date‘)  = curdate()
;

-- (2) How many redirects to Soundcloud from embedded music players?

SELECT     COUNT(‘re‘.‘re_url‘) AS ‘redirects‘
FROM       ‘SONG‘        AS ‘so‘
INNER JOIN ‘INTERACTION‘ AS ‘i‘
ON         ‘so‘.‘so_id‘  =  ‘i‘.‘m_id‘
INNER JOIN ‘REDIRECT‘    AS ‘re‘
ON         ‘i‘.‘i_id‘    =  ‘re‘.‘re_id‘
;

-- (3) What is the redirect rate for videos?

SELECT          COUNT(‘pl‘.‘pl_id‘) AS ‘plays‘
    ,           COUNT(‘re‘.‘re_id‘) AS ‘redirects‘
FROM            ‘MUSIC_VIDEO‘ AS ‘mv‘
INNER JOIN      ‘INTERACTION‘ AS ‘i‘
ON              ‘mv‘.‘mv_id‘  =  ‘i‘.‘m_id‘
LEFT OUTER JOIN ‘PLAY‘        AS ‘pl‘
ON              ‘i‘.‘i_id‘    =  ‘pl‘.‘pl_id‘
LEFT OUTER JOIN ‘REDIRECT‘    AS ‘re‘
ON                ‘i‘.‘i_id‘  =  ‘re‘.‘re_id‘
;

-- (4) Is Black Silk Hooded Sweatshirt sold out?

SELECT     ‘pg‘.‘pg_quantity_available‘ > 0 AS ‘is_available‘
FROM       ‘PHYSICAL_GOOD‘ AS ‘pg‘
INNER JOIN ‘GOOD‘ AS ‘g‘
ON         ‘pg‘.‘pg_id‘ = ‘g‘.‘g_sku‘
WHERE      ‘g‘.‘g_name‘ = "Black Silk Hooded Sweatshirt"
;

-- (5) How many Red Summer Beanie items were sold in October 2014?

SELECT count(‘s‘.‘sale_id‘) AS ‘sold‘
FROM  ‘PURCHASE‘      AS ‘s‘
```

```
WHERE ‘s‘.‘sale_date‘ >= "2014-10-1"
AND   ‘s‘.‘sale_date‘ <= "2014-10-31"
;

-- (6) What digital goods are currently unavailable?

SELECT     ‘g‘.‘g_name‘
FROM       ‘DIGITAL_GOOD‘         AS ‘dg‘
INNER JOIN ‘GOOD‘                 AS ‘g‘
ON         ‘dg‘.‘dg_id‘           =  ‘g‘.‘g_sku‘
WHERE      ‘dg‘.‘dg_is_available‘ =  0
;

-- (7) How many orders do I have to fill to the US?

SELECT COUNT(‘sale‘.‘sale_id‘)  AS ‘orders‘
FROM       ‘PURCHASE‘           AS ‘sale‘
INNER JOIN ‘PHYSICAL_CONSUMER‘  AS ‘pc‘
ON         ‘sale‘.‘c_id‘        =  ‘pc‘.‘pc_id‘
WHERE      ‘pc‘.‘pc_country‘    =  "US"
AND        ‘sale‘.‘sale_status‘ =  "recieved"
;

-- (8) What precentage of my digital consumers are from outside the US?

SELECT COUNT(‘dc‘.‘dc_id‘)      AS ‘total‘
    ,  COUNT(‘foreign‘.‘dc_id‘) AS ‘foreign‘
FROM ‘DIGITAL_CONSUMER‘         AS ‘dc‘
LEFT OUTER JOIN (
SELECT ‘dc_id‘
FROM   ‘DIGITAL_CONSUMER‘
    WHERE  ‘dc_country‘ != "US"
) AS ‘foreign‘
ON ‘dc‘.‘dc_id‘ = ‘foreign‘.‘dc_id‘
;

-- (9) How many digital consumers are also physical consumers?

SELECT COUNT(‘dc‘.‘dc_id‘) AS ‘digital_and_physical‘
FROM ‘DIGITAL_CONSUMER‘ AS ‘dc‘
INNER JOIN ‘PHYSICAL_CONSUMER‘ AS ‘pc‘
ON ‘dc‘.‘dc_id‘ = ‘pc‘.‘pc_id‘
;

-- (10) What products have garnered zero sales in the past 14 days?

SELECT ‘g‘.‘g_name‘
FROM   ‘GOOD‘                     AS ‘g‘
LEFT JOIN (
```

```
SELECT     ‘li‘.‘g_sku‘       AS ‘g_sku‘
FROM       ‘PURCHASE‘        AS ‘sale‘
INNER JOIN ‘LINE_ITEM‘       AS ‘li‘
ON         ‘li‘.‘sale_id‘    =  ‘sale‘.‘sale_id‘
WHERE      ‘sale‘.‘sale_date‘ >= DATE_SUB(curdate(), INTERVAL 2 WEEK)
)   AS         ‘items_sold‘
ON     ‘g‘.‘g_sku‘               = ‘items_sold‘.‘g_sku‘
WHERE  ‘items_sold‘.‘g_sku‘      IS NULL
;
```

VI.3 Design Tradeoffs and Limitations

Music videos and songs have some duplicate fields that could be added to the parent Media table. This was done to allow individual constraints to be placed upon those fields. For example there is both an 'so_url' and a 'mv_url', the separation here would allow for specific constraints to be placed, like checking that "youtube" is somewhere in the link of a video.

Chapter VII

Database Integrity and Security

VII.1 Functional Dependencies

Media Dependencies

1. m_id ⟶ m_upload_date, so_id, mv_id
2. so_id ⟶ so_title, so_artist, so_url, m_id
3. mv_id ⟶ mv_title, mv_url, m_id
4. i_id ⟶ i_date, m_id, re_id, pl_id
5. re_id ⟶ re_url, i_id
6. pl_id ⟶ i_id

Sales Dependencies

1. c_id ⟶ c_firstname, c_lastname, c_email, c_phone, dc_id, pc_id
2. dc_id ⟶ dc_country, c_id
3. pc_id ⟶ pc_address_line_1, pc_address_line_2, pc_country, pc_state, c_id
4. sale_id ⟶ sale_status, sale_date, sale_fulfill_date
5. g_sku ⟶ g_name, g_description, g_price, pg_id, dg_id
6. pg_id ⟶ pg_color, pg_size, pg_quantity_available, g_sku
7. dg_id ⟶ dg_is_available, g_sku
8. (sale_id, g_sku) ⟶ li_quantity

VII.2 Adjustments for Normalization

John's database has already been presented in third normal form. No further normalization is necessary.

VII.3 Integrity and Security

- All identifiers must should be generated by the dbms
- All date fields should be generated in a consistent manner across all records
- A valid email regex should be used to ensure that supplied email addresses are at least syntactically valid
- **sale_status** must be one of “received”, “shipped” or “fulfilled”
- **sale_date** should be prior to **sale_fulfill_date**
- **li_quantity** should be non-negative
- **g_price** should be non-negative
- **pc_state** should be a standard US 2-character state code
- **i_date** should be prior to **m_upload_date**
- **mv_url, so_url, re_url** should be verified by a regex to ensure that they are valid urls

Chapter VIII

Implementation Notes

VIII.1 Indices

An index should be placed on **c_id** as it is the primary index of customers. Having this as a primary, physical index will improve search times for a customer by id. This will reduce the operation time of join operations, such as the join to figure out which purchase belongs to a specific customer.

A primary index should be placed on **sale_id** and **g_sku** as physical indexes here will improve search times for individual elements. This will also make the joins between Purchase and Good through Line_Item quicker.

No further indexes will be placed. The main information that John wants to see is available through joining tables together. Since physical indexes have already been placed on the join-on properties, these tables have been indexed as effectively as necessary.

VIII.2 Data

```
INSERT INTO CUSTOMER (c_id, c_firstname, c_lastname, c_email, c_phone)
VALUES ('cust0000000000000001', 'Jobe', 'Lau', 'jlau@hotmail.com
    ', '555-402-2023')
     , ('cust0000000000000002', 'Rodney', 'Elway', 'r.j.elway@gmail.com
        ', '555-231-2415')
     , ('cust0000000000000003', 'Missy', 'Moore', 'ladiva@gmail.com
        ', '555-235-2425')
     , ('cust0000000000000004', 'Jam', 'Boree', 'jb@business.io
        ', '555-215-5232')
     , ('cust0000000000000005', 'Martin', 'Garnier', 'm.garnier@garnier
        .org', '555-250-9439')
;

INSERT INTO DIGITAL_CONSUMER (dc_id, dc_country)
VALUES ('cust0000000000000001', 'Albania')
     , ('cust0000000000000002', 'US')
     , ('cust0000000000000003', 'US')
     , ('cust0000000000000004', 'Germany')
;

INSERT INTO PHYSICAL_CONSUMER (pc_id, pc_address_line_1,
    pc_address_line_2, pc_country, pc_state)
VALUES ('cust0000000000000003', '123 Travesty Ln.', '', 'US', 'TX')
     , ('cust0000000000000005', '600 Alra Blvd.', 'Unit #545', 'US', 'UT
        ')
;

INSERT INTO PURCHASE (sale_id, sale_status, sale_fulfill_date,
    sale_date, c_id)
VALUES ('sale0000000000000001', 'shipped', '2014-10-02
    07:31:52-08:00', '2014-10-01 12:13:57-08:00', 'cust0000000000000003
    ')
     , ('sale0000000000000002', 'recieved', 'NULL', '2014-12-03
        12:17:29-08:00', 'cust0000000000000005')
     , ('sale0000000000000003', 'fulfilled', '2014-12-07
        09:11:51-08:00', '2014-12-07 09:11:49-08:00', '
        cust0000000000000001')
     , ('sale0000000000000004', 'fulfilled', '2014-12-07
        11:08:18-08:00', '2014-12-07 11:08:17-08:00', '
        cust0000000000000002')
     , ('sale0000000000000005', 'fulfilled', '2014-12-07
        12:11:52-08:00', '2014-12-07 12:11:50-08:00', '
        cust0000000000000003')
     , ('sale0000000000000006', 'fulfilled', '2014-12-08
        01:12:40-08:00', '2014-12-08 01:12:39-08:00', '
        cust0000000000000004')
;
```

```
INSERT INTO GOOD (g_sku,g_name,g_description,g_price)
VALUES ('good0000000000000001','Rise of John: Rap Emperor','The
    first mixtape by John. 2007. Bone Records.','0')
     , ('good0000000000000002','Return of John: No more Disciples','
        After 6 years John is back, with hot new beats and tracks
        featuring the latest and greates. 2013. Bone Records.','0')
     , ('good0000000000000003','Black Silk Hooded Sweatshirt','
        Quality Egyptian wool-silk, unique black dye patterns for a
        trendy look.','2499')
     , ('good0000000000000004','Red Summer Beanie','A comfortable
        beanie for the summertime','1299')
     , ('good0000000000000005','Plain White Perfect Tee','The
        perfect white t-shirt you have always wanted. 100% Cotton and
         woven with care.','999')
;

INSERT INTO PHYSICAL_GOOD (pg_id,pg_color,pg_size,
    pg_quantity_available)
VALUES ('good0000000000000003','Black','M','3')
     , ('good0000000000000004','Red','OSFA','7')
     , ('good0000000000000005','White','M','2')
;

INSERT INTO DIGITAL_GOOD (dg_id,dg_is_available)
VALUES ('good0000000000000001',0)
     , ('good0000000000000002',1)
;

INSERT INTO LINE_ITEM (li_quantity,sale_id,g_sku)
VALUES ('2','sale0000000000000001','good0000000000000004')
     , ('1','sale0000000000000002','good0000000000000004')
     , ('1','sale0000000000000002','good0000000000000005')
     , ('1','sale0000000000000003','good0000000000000001')
     , ('1','sale0000000000000004','good0000000000000002')
     , ('1','sale0000000000000005','good0000000000000001')
     , ('1','sale0000000000000006','good0000000000000002')
;

INSERT INTO MEDIA (m_id,m_upload_date)
VALUES ('media000000000000001','2014-10-15 13:23:31-08:00')
     , ('media000000000000002','2014-10-30 18:12:42-08:00')
     , ('media000000000000003','2014-11-2 08:23:09-08:00')
     , ('media000000000000004','2014-11-2 08:30:12-08:00')
     , ('media000000000000005','2014-11-27 11:45:10-08:00')
     , ('media000000000000006','2014-12-6 09:03:19-08:00')
;

INSERT INTO SONG (so_id,so_title,so_artist,so_url)
```

```
VALUES ('media000000000000002','Bad Phone','John','soundcloud.com/
    john/song/22')
     , ('media000000000000003','Street Judgement ft. Lawriee','John
        ','soundcloud.com/john/song/23')
     , ('media000000000000006','Walk Before You Crawl ft. Caleb','
        John','soundcloud.com/john/song/45')
;

INSERT INTO MUSIC_VIDEO (mv_id,mv_title,mv_url)
VALUES ('media000000000000001','A day in the life of John pt. 1','
    youtube.com/v/aExs42')
     , ('media000000000000004','Street Judgement [OFFICIAL VIDEO]','
        youtube.com/v/820uykY')
     , ('media000000000000005','Leading the Way [OFFICIAL VIDEO]','
        youtube.com/v/w92n212')
;

INSERT INTO INTERACTION (i_id,i_date,m_id)
VALUES ('inter000000000000001','2014-12-10 11:47:23-08:00','
    media000000000000001')
     , ('inter000000000000002','2014-12-12 13:08:20-08:00','
        media000000000000001')
     , ('inter000000000000003','2014-12-12 13:47:23-08:00','
        media000000000000002')
     , ('inter000000000000004','2014-12-13 08:02:07-08:00','
        media000000000000006')
     , ('inter000000000000005','2014-12-13 09:44:21-08:00','
        media000000000000003')
     , ('inter000000000000006','2014-12-14 13:57:13-08:00','
        media000000000000003')
     , ('inter000000000000007','2014-12-14 14:08:03-08:00','
        media000000000000004')
     , ('inter000000000000008','2014-12-15 15:59:49-08:00','
        media000000000000002')
     , ('inter000000000000009','2014-12-15 18:42:03-08:00','
        media000000000000006')
     , ('inter000000000000010','2014-12-15 19:02:02-08:00','
        media000000000000005')
     , ('inter000000000000011','2014-12-15 23:41:48-08:00','
        media000000000000005')
     , ('inter000000000000012','2014-12-16 07:21:15-08:00','
        media000000000000005')
     , ('inter000000000000013','2014-12-17 08:31:18-08:00','
        media000000000000005')
     , ('inter000000000000014','2014-12-18 19:20:26-08:00','
        media000000000000005')
;

INSERT INTO PLAY (pl_id)
```

```
VALUES ('inter000000000000001')
     , ('inter000000000000003')
     , ('inter000000000000004')
     , ('inter000000000000005')
     , ('inter000000000000007')
     , ('inter000000000000008')
     , ('inter000000000000010')
     , ('inter000000000000011')
     , ('inter000000000000012')
     , ('inter000000000000013')
     , ('inter000000000000014')
;

INSERT INTO REDIRECT (re_id,re_url)
VALUES ('inter000000000000002','youtube.com/v/aExs42')
     , ('inter000000000000006','soundcloud.com/john/song/23')
     , ('inter000000000000009','soundcloud.com/john/song/45')
;
```

VIII.3 Query Trace

A bug was occuring for me in phpMyAdmin. To do the query trace a separate window is opened up, when I attempt to load in the files there phpMyAdmin ALWAYS loses my login token. As a result I cannot see or generate a query trace. The history view cannot be accessed outside of the popup menu. I have tried several browsers, operating systems and orders of doing this process, to no avail.

I can confirm that creating the database, inserting data values, and running my set of queries all worked successfully.

VIII.4 Implementation Assessment

The order that I went about things really helped my implementation. After several iterations of improving the design of my database I had my ERB/tables all set and correct. This was essential in being able to generate data that would reveal the success of my queries. When writing data and queries it is important that all the naming and conventions be out of the way.

Having a solid ERB is probably the most important aspect of implementation. It is the best point of reference for doing mock data, queries as well as the generation of the database which I used Ebarcadero Studio for, which I found to be essential. Using tools was important for minimizing mistakes, the SQL compiler alerted me to any other issues that were missed.

The most difficult part of implementation was trying to get the query trace to work. I didn't know how to do it initially, and with internet help it seems that there is a bug in phpMyAdmin that prevented me from gathering the trace data for that section.

Chapter IX

Lessons Learned

By the end of the project I felt that I was using all the right tools. Using LaTeXand ER Studio were the two most essential tools. LaTeXallowed me to compose my project in a programmatic way. It automatically respects changes to all source files such as images and raw sql files. ER Studio made designing the ERB simple and the database generation came for free once the ERB was correct. Like LaTeX, ER Studio allowed me to only make changes in one place, reducing the element of human error.

I used as many LaTeXhelpers as I could, I had a site that converted excel tables into latex tables. I would simply do my table work in Google Sheets (excel in Google Docs) then use the tool to do most of the work to translate. I incorporated LaTeXpackages as necessary. I used longtable for multi-page tables, verbatim to include textual code files, and several more included and commented-on in this tex file.

I do not know how many hours per week I spent on the project, it must have been at least 6-8. The time I spent was often clumped up into two to three days per deliverable. It would have been better to split the work out—in retrospect a lot of the sections of this project can have meaningful work done towards them in a single hour or less. What helped me tremendously was the feedback and office hours. Through them I was able to correct the mistakes I was making and come to a better understanding of relational database systems. Designing my database as correctly as possible was a big priority for me, and as the semester went on my database actually became simpler and easier to use as a tool.

The project certainly helped me learn SQL. More importantly it helped me understand what SQL was for me to write. Database generation should be done by ER Studio. Data generation should be drafted in a spreadsheet then transpiled into SQL statements later. Only the queries themselves should be written by hand. I think that we could have learned ERB's slightly earlier in the semester to have our own database ERB's done sooner because they were so helpful. Even if they were bad or horribly incorrect, using ER Studio they can easily be iterably improved with feedback and question-asking. I got a good sense of normalization and how it is one of those things that 'just happens' when a database is designed correctly. I found that the better the design was the simpler the normalization looked.

MySQL was a good DBMS to use, it has the most normal SQL syntax compared to MS SQL and it isn't Oracle. The popularity of MySQL made it easy to get query help and debug why things weren't working. If I had to use another relational DBMS, I would be very interested in using Postgres. I hear that MySQL is not the best representative of its kind, despite being the most popular.