

Modeling an Online Music Business

Zane Kansil
Loyola Marymount University
Database Systems

December 3, 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	8
IV.3	User-oriented data dictionary	9
IV.4	Cross-reference table	11
V	Enterprise Database Design	13
V.1	Logical model of the Enterprise	13
V.1.1	List of Entities and Attributes	13
V.1.2	List of Relationships and Attributes	15
V.1.3	Entity-Relationship diagram of the Enterprise	16
V.2	Conceptual model of the enterprise	17
V.3	Table dictionary	19
V.4	Attribute dictionary	20
VI	Database and Query Definition	21
VI.1	Database Definition	21
VI.2	Database Queries	30
VI.3	Design Tradeoffs and Limitations	32
VII	Database Integrity and Security	33
VII.1	Functional Dependencies	33
VII.2	Adjustments for Normalization	33
VII.3	Integrity and Security	33
VIII	Implementation Notes	34
VIII.1	Indices	34
VIII.2	Data	34
VIII.3	Query Trace	34
VIII.4	Implementation Assessment	34
IX	Lessons Learned	35

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 physical goods are currently frozen? (sales prevented)
7. How many orders do I have to fill to the US?
8. What percentage of my digital consumers are from outside the US?
9. What is the average monthly revenue over the past six months?
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
- Upload/Choose an Image
- Color (optional)
- Size (optional)
- Price (in USD)
- Current stock on hand

4. Add New Digital good

- Name
- Description Paragraph
- Upload/Choose an Image
- Price (in USD)

5. Physical Good Admin View

- Name
- Good SKU
- Good description (editable)
- Good's image url (editable)
- Price (in USD)
- Color of Good (editable)
- Quantity
- Size of Good

6. Digital Good Admin View

- Name
- Good SKU
- Good description (editable)
- Good's image url (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 preferred 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	Computer-assigned identifier
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.
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_image_url	URL of image for 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
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.
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_upload_date	Datetime that the video was uploaded to 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_state	State which customer resides
pc_zip_code	Zip code. 5 digits in the US.
pg_color	Color of 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)

redirect_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	Datetime this sale was set to 'fulfilled'
sale_id	Sales/transaction identifier
sale_status	The shipping status, 'recieved', '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_upload_date	Datetime that the video was uploaded to SoundCloud
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							x	x		
c_last_name							x	x		
c_phone							x	x		
dc_country								x		
dg_is_available				x		x			x	
g_description			x	x	x	x				
g_image_url			x	x	x	x				
g_name			x	x	x	x			x	
g_price			x	x	x	x			x	
g_sku					x	x			x	
i_date										x
li_number									x	
li_quantity									x	
m_id										x
mv_plays	x									
mv_plays_today	x									
mv_redirects	x									
mv_title	x									x
mv_upload_date	x									
mv_url	x									
pc_address_line_1							x			
pc_address_line_2							x			
pc_country							x			
pc_state							x			
pc_zip_code							x			
pg_color			x		x				x	
pg-quantity_available			x		x					
pg_size			x		x				x	
redirect_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_upload_date		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_title
- mv_url (url video is hosted at)
- mv_upload_date

3. Song

- so_title
- so_artist
- so_url (url song is hosted at)
- so_upload_date

4. Play

- i_date

5. Redirect

- i_date
- redirect_url (“YouTube” or “SoundCloud”)

6. Consumer

- c_id
- c_firstname
- c_lastname
- c_email
- c_phone

7. Physical Consumer

- pc_address_line_1

- pc_address_line_2
- pc_country
- pc_state

8. Digital Consumer

- dc_country (optional)

9. Purchase

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

10. Line Item

- li_number
- li_quantity

11. Good

- g_sku
- g_name
- g_description
- g_price

12. Digital Good

- dg_is_available (boolean, used to prevent ordering)

13. Physical Good

- 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

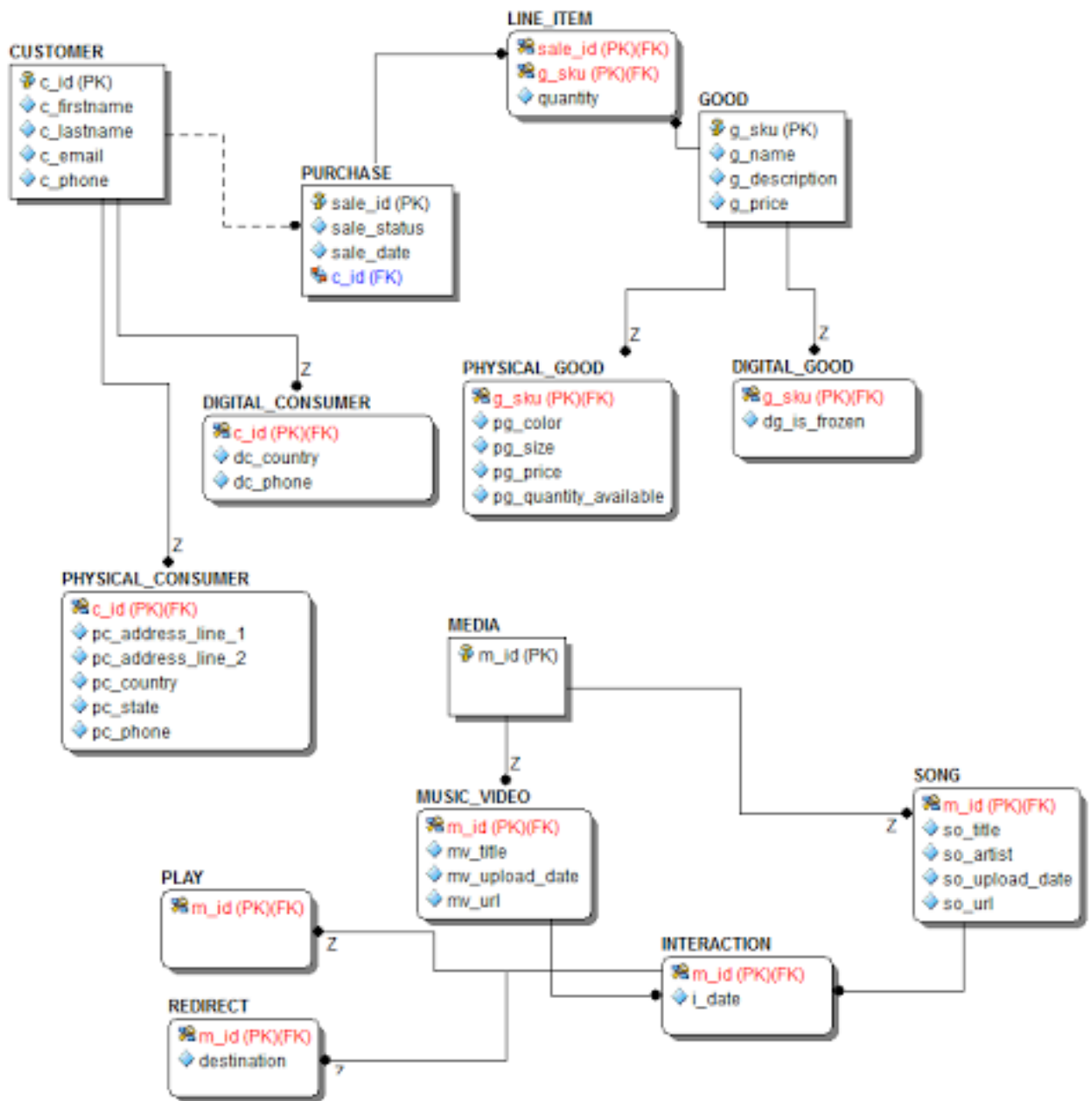
Media Relationships

1. MUSIC_VIDEO has many INTERACTION
2. INTERACTION has 1 MUSIC_VIDEO
3. SONG has many INTERACTION
4. INTERACTION has 1 SONG
5. INTERACTION is a PLAY
6. INTERACTION is a REDIRECT

Sales Relationships

1. CONSUMER is a PHYSICAL_CONSUMER
2. CONSUMER is a DIGITAL_CONSUMER
3. CONSUMERS make many PURCHASE
4. PURCHASE has 1 CUSTOMER
5. PURCHASE has many LINE_ITEM
6. PURCHASE has 1 STATUS
7. STATUS is applicable to 1 PURCHASE
8. LINE_ITEM belongs to 1 PURCHASE
9. LINE_ITEM has 1 GOOD
10. GOOD can appear in many LINE_ITEM
11. GOOD is a DIGITAL_GOOD
12. GOOD is a PHYSICAL_GOOD

V.1.3 Entity-Relationship diagram of the Enterprise



-ZK: will upgrade resolution next time...

V.2 Conceptual model of the enterprise

MEDIA(m_id)

MUSIC_VIDEO(m_id, mv_title, mv_url, mv_upload_date)

PK/FK: m_id

CK: m_id, mv_title, mv_url

SONG(m_id, so_title, so_artist, so_url, so_upload_date)

PK/FK: m_id

CK: m_id, so_title, so_url

PLAY(m_id, i_datetime)

PK/FK: m_id

CK: m_id

REDIRECT(m_id, m_to)

PK/FK: m_id

CK: m_id

CONSUMER(c_id, c_firstname, c_lastname, c_email)

PK: c_id

CK: c_id, c_email

PHYSICAL_CONSUMER(
 c_id

, pc_address_line_1

, pc_address_line_2

, pc_country

, pc_state

, pc_phone

)

PK/FK: c_id

CK: c_id, pc_phone

DIGITAL_CONSUMER(
 c_id

, pc_phone

, pc_country

)

PK/FK: c_id

CK: c_id, pc_phone

PURCHASE(p_id, c_id, p_date)

PK/FK: p_id

CK: p_id, c_id

LINE_ITEM(p_id, g_id, quantity)

PK: p_id

CK: p_id

FK: p_id, g_id

GOOD(g_name, g_description, g_sku, g_price)

PK: g_sku

CK: g_name, g_sku

DIGITAL_GOOD(g_sku, dg_is_frozen)

PK: g_sku

CK: g_sku

PHYSICAL_GOOD(g_sku, pg_color, pg_size, pg_quantity_available)

PK/FK: g_sku

CK: g_sku

V.3 Table dictionary

V.4 Attribute dictionary

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, November 04, 2014 21:18:16
-- Target DBMS : MySQL 5.x
--
```

```
--
-- TABLE: CUSTOMER
--
```

```
CREATE TABLE CUSTOMER(
    c_id          VARCHAR(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(
    c_id          VARCHAR(20)    NOT NULL,
    dc_country    VARCHAR(20)    NOT NULL,
    dc_phone      VARCHAR(20)    NOT NULL,
    PRIMARY KEY (c_id)
)ENGINE=INNODB
```

```

;

--
-- TABLE: DIGITAL_GOOD
--

CREATE TABLE DIGITAL_GOOD(
    g_sku          VARCHAR(20)    NOT NULL,
    dg_is_frozen   BIT(1)         NOT NULL,
    PRIMARY KEY (g_sku)
)ENGINE=INNODB
;

--
-- TABLE: GOOD
--

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

--
-- TABLE: INTERACTION
--

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

--
-- TABLE: LINE_ITEM
--

```

```

CREATE TABLE LINE_ITEM(
    sale_id    VARCHAR(20)    NOT NULL,
    g_sku      VARCHAR(20)    NOT NULL,
    quantity   INT,
    PRIMARY KEY (sale_id, g_sku)
)ENGINE=INNODB
;

--
-- TABLE: MEDIA
--

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

--
-- TABLE: MUSIC_VIDEO
--

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

--
-- TABLE: PHYSICAL_CONSUMER
--

CREATE TABLE PHYSICAL_CONSUMER(
    c_id      VARCHAR(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,
    pc_phone     VARCHAR(20)    NOT NULL,
    PRIMARY KEY (c_id)
)ENGINE=INNODB
;

```

```

)ENGINE=INNODB
;

--
-- TABLE: PHYSICAL_GOOD
--

CREATE TABLE PHYSICAL_GOOD(
    g_sku          VARCHAR(20)    NOT NULL,
    pg_color       VARCHAR(20),
    pg_size        VARCHAR(20),
    pg_price       FLOAT(8, 0)    NOT NULL,
    pg_quantity_available INT      NOT NULL,
    PRIMARY KEY (g_sku)
)ENGINE=INNODB
;

--
-- TABLE: PLAY
--

CREATE TABLE PLAY(
    m_id          VARCHAR(20)    NOT NULL,
    PRIMARY KEY (m_id)
)ENGINE=INNODB
;

--
-- TABLE: PURCHASE
--

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

--

```



```
-- TABLE: REDIRECT
--
```

```
CREATE TABLE REDIRECT(
  m_id          VARCHAR(20)    NOT NULL,
  destination    VARCHAR(20)    NOT NULL,
  PRIMARY KEY (m_id)
)ENGINE=INNODB
;
```

```
--
-- TABLE: SONG
--
```

```
CREATE TABLE SONG(
  m_id          VARCHAR(20)    NOT NULL,
  so_title       VARCHAR(50)    NOT NULL,
  so_artist      VARCHAR(20)    NOT NULL,
  so_upload_date DATETIME       NOT NULL,
  so_url         VARCHAR(100)   NOT NULL,
  PRIMARY KEY (m_id)
)ENGINE=INNODB
;
```

```
--
-- INDEX: Ref22
--
```

```
CREATE INDEX Ref22 ON DIGITAL_CONSUMER(c_id)
;
```

```
-- INDEX: Ref97
--
```

```
CREATE INDEX Ref97 ON DIGITAL_GOOD(g_sku)
;
```

```
-- INDEX: Ref148
--
```

```
CREATE INDEX Ref148 ON INTERACTION(m_id)
;
```

```
-- INDEX: Ref84
--
```

```

CREATE INDEX Ref84 ON LINE_ITEM(sale_id)
;
--
-- INDEX: Ref95
--

CREATE INDEX Ref95 ON LINE_ITEM(g_sku)
;
--
-- INDEX: Ref1610
--

CREATE INDEX Ref1610 ON MUSIC_VIDEO(m_id)
;
--
-- INDEX: Ref21
--

CREATE INDEX Ref21 ON PHYSICAL_CONSUMER(c_id)
;
--
-- INDEX: Ref96
--

CREATE INDEX Ref96 ON PHYSICAL_GOOD(g_sku)
;
--
-- INDEX: Ref1312
--

CREATE INDEX Ref1312 ON PLAY(m_id)
;
--
-- INDEX: Ref23
--

CREATE INDEX Ref23 ON PURCHASE(c_id)
;
--
-- INDEX: Ref1313
--

CREATE INDEX Ref1313 ON REDIRECT(m_id)
;
--
-- INDEX: Ref1611
--

```

```

CREATE INDEX Ref1611 ON SONG(m_id)
;
--
-- TABLE: DIGITAL_CONSUMER
--

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

--
-- TABLE: DIGITAL_GOOD
--

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

--
-- TABLE: INTERACTION
--

ALTER TABLE INTERACTION ADD CONSTRAINT RefMUSIC_VIDEO8
    FOREIGN KEY (m_id)
    REFERENCES MUSIC_VIDEO(m_id)
;

ALTER TABLE INTERACTION ADD CONSTRAINT RefSONG9
    FOREIGN KEY (m_id)
    REFERENCES SONG(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 (m_id)
    REFERENCES MEDIA(m_id)
;

--
-- TABLE: PHYSICAL_CONSUMER
--

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

--
-- TABLE: PHYSICAL_GOOD
--

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

--
-- TABLE: PLAY
--

ALTER TABLE PLAY ADD CONSTRAINT RefINTERACTION12
    FOREIGN KEY (m_id)
    REFERENCES INTERACTION(m_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 (m_id)
    REFERENCES INTERACTION(m_id)
;

--
-- TABLE: SONG
--

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

```

VI.2 Database Queries

-- (1) How many video plays today?

```
SELECT count('p'.'i_date')
FROM      'MUSIC_VIDEO' AS 'mv'
INNER JOIN 'PLAY'      AS 'p'
ON        'mv'.'m_id'   = 'p'.'m_id'
WHERE     'p'.'i_date'  >= curdate()
AND      'p'.'i_date'  <= curdate()
;
```

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

```
SELECT count('r'.'to')
FROM      'SONG'        AS 's'
INNER JOIN 'REDIRECT'   AS 'r'
ON        's'.'m_id'    = 'r'.'m_id'
WHERE     'r'.'to'      = "soundcloud"
;
```

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

```
SELECT count('p'.'m_id'), count('r'.'m_id')
FROM      'MUSIC_VIDEO' AS 'mv'
INNER JOIN 'PLAY'      AS 'p'
ON        'mv'.'m_id'   = 'p'.'m_id'
INNER JOIN 'REDIRECT'   AS 'r'
ON        'mv'.'m_id'   = 'r'.'m_id'
;
```

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

```
SELECT 'pg'.'pg_quantity_available'
FROM    'PHYSICAL_GOOD' AS 'pg'
;
```

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

```
SELECT count('s'.'sale_id')
FROM    'PURCHASE'      AS 's'
WHERE   's'.'sale_date' >= "2014-10-1"
AND     's'.'sale_date' <= "2014-10-31"
;
```

-- (6) What physical goods are currently frozen? (sales prevented)

```
SELECT 'pg'.'pg_name'
FROM    'PHYSICAL_GOOD' AS 'pg'
```

```
WHERE 'pg'.'pg_is_frozen' = 1
;
```

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

```
SELECT count('pg'.'sale_id')
FROM      'PHYSICAL_CUSTOMER' AS 'pc'
INNER JOIN 'PURCHASE'          AS 'sale'
ON        'pc'.'c_id'          = 'sale'.'c_id'
WHERE     'pc'.'pc_country'    = "US"
;
```

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

```
SELECT count('dg'.'c_id'), count('foreign'.'c_id')
FROM    'DIGITAL_CUSTOMER'      AS 'dg'
,       'DIGITAL_CUSTOMER'      AS 'foreign'
WHERE   'foreign'.'dc_country' != "US"
;
```

-- (9) What is the average monthly revenue over the past six months?

INCOMPLETE

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

```
SELECT DISTINCT 'g'.'g_name'
FROM            'LINE_ITEM'      AS 'li'
LEFT JOIN      'GOOD'           AS 'g'
ON             'li'.'g_sku'      = 'g'.'g_sku'
WHERE          'li'.'sale_date' >= DATE_SUB(curdate(), INTERVAL 2 WEEK)
AND           'g'.'g_sku'       IS NULL
;
```

VI.3 Design Tradeoffs and Limitations

Not too many limitation currently. I recently added a parent **MEDIA** entity for videos and songs.

Chapter VII

Database Integrity and Security

VII.1 Functional Dependencies

A list of the functional dependencies that hold on your database.

VII.2 Adjustments for Normalization

An explanation of the changes needed to normalize your database.

VII.3 Integrity and Security

A list (in English) of the integrity and security constraints which are to hold on your database.

Chapter VIII

Implementation Notes

VIII.1 Indices

A list of the indices used by your database, with a justification for each.

VIII.2 Data

The data used to populate your database.

VIII.3 Query Trace

A trace of the execution of each of your queries.

VIII.4 Implementation Assessment

An assessment of how smoothly your implementation went

Chapter IX

Lessons Learned