#### Cover Sheet

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UNIT CODE:**  HDB201  **TITLE:**  Database Development | | **STUDENT/PROJECT/TEAM NAME**  Man Fu Lei | | | |
| **NAME OF LECTURER**  Stewart Godwin | | | | **DUE DATE**  October 31, 2014 | |
| **TOPIC OF ASSIGNMENT**  Database Development Project | | | | | |
| **Group or tutorial**  *(if applicable)*  Group | **Course**  Associate Degree of Network Technology | | | **Campus**  Thornlie | |
| I certify that the attached assignment is my own work and that any material drawn from other sources has been acknowledged.  Copyright in assignments remains my property. I grant permission to Polytechnic West to make copies of assignments for assessment, review and/or record keeping purposes. I note that Polytechnic West reserves the right to check my assignment for plagiarism. Should the reproduction of all or part of an assignment be required by Polytechnic West for any purpose other than those mentioned above, appropriate authorisation will be sought from me on the relevant form. | | | | | |
| This assessment is submitted as a group in accordance with the guidelines for the assessment in this subject. If handing in an assignment in a paper or other physical form, sign here to indicate that you have read this form, filled it in completely and that you certify as above. | | | | | |
| **Signature** | | | **Student ID Number** | | **Date** |
| Man Fu Lei | | | 131306105 | | October 30, 2014 |
| Isaac Khoza | | | 131104643 | | October 30, 2014 |
| Ugyen Dema | | | 131314861 | | October 30, 2014 |
|  | | |  | |  |
|  | | |  | |  |
| OR, if submitting this paper electronically as per instructions for the unit, place an ‘x’ in the box below to indicate that you have read this form and filled it in completely and that you certify as above. Please include this page in/with your submission. Any electronic responses to this submission will be sent to your Polytechnic West email address.  **Agreement**  **Date** October 30, 2014 | | | | | |

**PROCEDURES AND PENALITIES ON LATE ASSIGNMENTS**

**Please refer to Student Handbook, Higher Education Qualifications**

* A student who wishes to defer the submission of an assignment must apply to the lecturer in charge of the relevant unit or course for an extension of the time within which to submit the assignment.
* Where an extension is sought for the submission of an assignment the application must: be in writing – preferably before the due date: and clearly state the grounds on which deferral are sought.
* Assignments submitted after the normal or extended date without approval shall incur a penalty of loss of marks.

**ACADEMIC MISCONDUCT**

All forms of cheating, plagiarism or collusion are regarded seriously and could result in penalties including loss of marks, exclusion from the unit or cancellation of enrolment.

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

**ASSIGNMENT RECEIPT**

To be completed by the student if the receipt is required.

|  |  |  |
| --- | --- | --- |
| **UNIT** | **NAME OF STUDENT** | **STUDENT ID. NO.** |
| **NAME OF LECTURER** | | **RECEIVED BY** |
| **Topic of assignment** | | **DATE RECEIVED** |

#### Peer Assessment Form

**Gantt chart**



**Evaluation matrix**

|  |  |  |  |
| --- | --- | --- | --- |
| **Student Name** | | **Isaac Khoza** | **Ugyen Dema** |
| **Leadership**-Showed individual leadership in the form of preparation, enthusiasm, commitment, organization, and communication to the degree appropriate to the position within the group by taking initiative. | 20 | 20 | 20 |
| **Cooperation**-Willingness to work together to accomplish the job of the group. | 20 | 20 | 20 |
| **Communication**-Shared information with the group, particularly in written form. | 20 | 20 | 20 |
| **Participation**-Did the appropriate share of work. | 20 | 20 | 20 |
| **Attendance**-Present and on time for work. | 20 | 20 | 20 |
| **TOTAL** | | /100 | /100 |

**Participation scale**

|  |  |
| --- | --- |
| **Student Name** | **Participation** |
| **Man Fu Lei** | 40 |
| **Isaac Khoza** | 30 |
| **Ugyen Dema** | 30 |
| **Total** | /100 |

Student: Man Fu Lei

Student Number: 131306105

Unit Name: Database Development

Submission Date: October 30, 2014

**Game Database Development**

**Develop a database for Acme Game ltd.**

#### Table of Content

[1 Introduction 1](#_Toc400291757)

[1.1 The Authorization 1](#_Toc400291758)

[1.2 The Purpose 1](#_Toc400291759)

[1.3 The Scope 1](#_Toc400291760)

[1.4 Terms of Reference 1](#_Toc400291761)

[1.5 Acknowledgements 2](#_Toc400291762)

[2 Method 3](#_Toc400291763)

[2.1 Online Research 3](#_Toc400291765)

[2.2 Group Discussion 3](#_Toc400291766)

[2.3 Documents 3](#_Toc400291767)

[3 Task1: Requirement Analysis 4](#_Toc400291768)

[3.1 Requirement description 4](#_Toc400291770)

[3.2 Major entities 5](#_Toc400291775)

[3.2.1 Supplier 5](#_Toc400291777)

[3.2.2 Age group 5](#_Toc400291778)

[3.2.3 Gender 6](#_Toc400291779)

[3.2.4 Distribution 6](#_Toc400291780)

[3.2.5 Inventory 6](#_Toc400291781)

[3.2.6 Game played 7](#_Toc400291782)

[3.2.7 Customer 7](#_Toc400291783)

[3.2.8 Retail outlet 7](#_Toc400291784)

[3.2.9 Employee 8](#_Toc400291785)

[3.2.10 Sales invoice 8](#_Toc400291786)

[3.2.11 Sales item 8](#_Toc400291787)

[3.2.12 Manufacture 9](#_Toc400291788)

[3.2.13 Outsourcing invoice 9](#_Toc400291789)

[3.2.14 Production 9](#_Toc400291790)

[3.2.15 Promotion event 10](#_Toc400291791)

[3.2.16 Outsourced product 10](#_Toc400291792)

[4 Task2: Conceptual Database Model 11](#_Toc400291793)

[4.1 Entity Relationship diagram in first normal form 11](#_Toc400291795)

[4.2 Relationships explanation 11](#_Toc400291796)

[5 Task3: Logical Database Model 13](#_Toc400291797)

[5.1 Entity Relationship diagram in third normal form 13](#_Toc400291799)

[5.2 Explanation of new entities and relationships 15](#_Toc400291800)

[5.3 Data types and sizes 17](#_Toc400291801)

[6 Task4: Database 23](#_Toc400291802)

[6.1 SQL codes for creating tables 23](#_Toc400291804)

[6.2 SQL codes for inserting sample records 44](#_Toc400291805)

[6.3 Screenshots 54](#_Toc400291806)

[7 Task5: User Interface 55](#_Toc400291807)

[7.1 User interface design 55](#_Toc400291809)

[7.1.1 Forms 55](#_Toc400291810)

[7.1.2 Reports 56](#_Toc400291811)

[7.2 Database functions 57](#_Toc400291812)

[8 Task6: SQL Queries 59](#_Toc400291813)

[8.1 Sales reports 59](#_Toc400291815)

[8.1.1 Report by age group 59](#_Toc400291822)

[8.1.2 Report by gender 59](#_Toc400291823)

[8.2 Inventory reports 60](#_Toc400291824)

[8.2.1 Report by specific genre 60](#_Toc400291826)

[8.2.2 Report by specific price range 61](#_Toc400291827)

[8.2.3 Most profitable games (highest to lowest) 62](#_Toc400291828)

[8.2.4 Total sales by month and year 63](#_Toc400291829)

[8.2.5 Most successful retailer 63](#_Toc400291830)

[9 Recommendation 65](#_Toc400291831)

[10 Conclusions 66](#_Toc400291832)

[11 Appendix 67](#_Toc400291833)

[11.1 Screenshots for sample data inserted 67](#_Toc400291837)

[12 List of References 76](#_Toc400291838)

[13 Glossary 77](#_Toc400291839)

[14 Index 80](#_Toc400291840)

#### Table of Figures

[Figure 1 – E-R diagram in first normal form 11](#_Toc400626970)

[Figure 2 – E-R diagram in third normal form 15](#_Toc400626971)

[Figure 3 – Created tables 54](#_Toc400626972)

[Figure 4 – Sample query form 55](#_Toc400626973)

[Figure 5 – Sample report 56](#_Toc400626974)

[Figure 7 – Report by age group 59](#_Toc400626975)

[Figure 8 – Report by gender 60](#_Toc400626976)

[Figure 9 – Report by specific genre 61](#_Toc400626977)

[Figure 10 – Report by specific price range 62](#_Toc400626978)

[Figure 11 – List of most profitable games 62](#_Toc400626979)

[Figure 12 – Sales by month and year 63](#_Toc400626980)

[Figure 13 – Sub-query for most successful customers 64](#_Toc400626981)

[Figure 14 – Most successful retailer 64](#_Toc400626982)

[Figure 15 – tblAgeGroup 67](#_Toc400626983)

[Figure 16 – tblCity 67](#_Toc400626984)

[Figure 17 – tblCountry 67](#_Toc400626985)

[Figure 18 – tblCustomer 67](#_Toc400626986)

[Figure 19 – tblCustomerBusiness 68](#_Toc400626987)

[Figure 20 – tblDate 68](#_Toc400626988)

[Figure 21 – tblDepartment 68](#_Toc400626989)

[Figure 22 – tblDistribution 68](#_Toc400626990)

[Figure 23 – tblEmployee 69](#_Toc400626991)

[Figure 24 – tblGamePlayed 69](#_Toc400626992)

[Figure 25 – tblGameTested 69](#_Toc400626993)

[Figure 26 – tblGender 70](#_Toc400626994)

[Figure 27 – tblGenre 70](#_Toc400626995)

[Figure 28 – tblInventory 70](#_Toc400626996)

[Figure 29 – tblIPR 70](#_Toc400626997)

[Figure 30 – tblIPRType 71](#_Toc400626998)

[Figure 31 – tblManufacture 71](#_Toc400626999)

[Figure 32 – tblMediaOutlet 71](#_Toc400627000)

[Figure 33 – tblMonth 71](#_Toc400627001)

[Figure 34 – tblOutsourcedProduct 72](#_Toc400627002)

[Figure 35 – tblOutsourcingInvoice 72](#_Toc400627003)

[Figure 36 – tblPosition 72](#_Toc400627004)

[Figure 37 – tblProduction 72](#_Toc400627005)

[Figure 38 – tblPromotionEvent 72](#_Toc400627006)

[Figure 39 – tblRetailOutlet 73](#_Toc400627007)

[Figure 40 – tblSalesInvoice 73](#_Toc400627008)

[Figure 41 – tblSalesItem 73](#_Toc400627009)

[Figure 42 – tblStage 73](#_Toc400627010)

[Figure 43 – tblState 74](#_Toc400627011)

[Figure 44 – tblSuburb 74](#_Toc400627012)

[Figure 45 – tblSupplier 74](#_Toc400627013)

[Figure 46 – tblYear 75](#_Toc400627014)

#### Executive Summary

The game market is growing faster and faster day by day and many companies have earned a lot from it. One of them is Acme Game ltd. which has grown up to a big company in 16 years. However its database system is relatively behind the times making the cooperation of different departments impede the continuing profit of the company. This report analyzed the requirement of the company, proposed a new database system from relational database model to logical design, gave the actual codes in Structured Query Language (SQL) to build the new database, inserted sample records, tested sample queries in sales and inventory reports and proposed a possible user interface design. The report also described how the database should work and gave recommendation that the new system can solve the existing problems the company encountered. It also gave enough screenshots to demonstrate that the database works as required.

# Introduction

## The Authorization

On Week 8 our team received the project from Acme Game ltd.

Acme Game Company was established in 1998 and has grown up to a company with revenue of 8 million dollars every year from a small company initially designing only small range of electronic games. It has developed more than 34 products aiming at people of two age groups ranging from 9 to 18 years old. It sells products designed itself and protected by Intellectual Property Rights (IPR) and those whose licenses are owned by the suppliers. The product is designed first by its own designers and then outsourced by oversea manufacturers. Recently the cooperation of its departments has encountered problem because of the drawbacks of the existing database system. The management level thus seeks us for help.

## The Purpose

The purpose of the project is to analyze the requirement and design a sample database for Acme Game ltd. The codes for the actual database will be provided and sample queries will be performed in the purpose of test.

## The Scope

The report contains a description of the requirements, the list of major entities, Entity Relationship (E-R) diagrams in both first normal form and third normal form and their explanations, data types and sizes specified for attributes, SQL codes for database creation and inserting sample data, user interface design and sample queries involving sales reports and inventory reports. The codes are written and tested in MySQL workbench conforming to the standard of SQL: 200n. Screenshots for sample data inserted are also available in the Appendix. There will be screenshots following these sections showing the codes work as expected.

## Terms of Reference

The referencing in this report uses American Psychological Association (APA) format. Information collected in this project is only for internal evaluation purpose and no disclosure will be made to third party. The audience of this report is people with basic SQL language and system design knowledge background. If referencing this document, please include all the author's names as it is a collaborative work. Sample data input is only for test usage and completely fictional. Any duplication with reality is substantially coincidence.

## Acknowledgements

The author wants to thank these people who helped us a lot during the project:

Isaac Khoza*, Polytechnic West Australia*

Ugyen Dema*, Polytechnic West Australia*

Stewart Godwin*, Polytechnic West Australia*

# Method



## Online Research

The author went through several webpages about the games and IPRs when considering the establishment of the database.

## Group Discussion

The team members talked with one another in order to discover the requirement of the new system and its mechanism.

## Documents

The team members review the documents concerning the company to understand how the company works.

# Task1: Requirement Analysis



## Requirement description

The first requirement is inventory control. It requires the database to track the products stored in distributions located in different places. If the information about a product changes the system administrator should be aware of it. Adding, deleting and updating the information of a product should be made easily without efforts raised by redundancy. In conditions such as that the storage of a product is low or the IPR or license is expired, the administrator should be notified. When changing attributes of the product such as unit price, retail price and location, it should be made easily without linking to different tables.

The second requirement is sales and marketing updates. It requires the database to track the information of the products sold and the trend of the current market concerning the possible interest of the people in the age groups. The database thus should keep the information of sales invoices, genre and the age groups. In order to cooperate tightly these information must be readily reachable by both departments. These data should be kept separately however linked together by the product itself.

The third requirement is the information of the promotional campaigns. It requires the database to keep a table about the promotional campaign events and record their times, media types and durations. The event must also be managed by one employee who is in charge of it. The event is related to the product so it should have a link to the product table which is not the same one as those in the inventory because they should be still in third stage where the products are in promotional stage. The cost of the event should be recorded too for reference of whether the event is successful or not.

The fourth requirement is the sales statistics. The database must provide functions of showing the report by sorting out the sales information about month, year, date, customer, place, product type and employee. While the product type and employee information are kept separately, the time related information should be kept with the invoice recording when it takes place. The invoice should also relate to the customer's information. The quantity of each product in each transaction should be recorded correctly too.

The fifth requirement is the development schedules. In order to monitor the development progress of the product, the start date, planned days for the stage and the completed percentage of current stage must be recorded. Because the process of developing a product requires four stages, the stage itself must also be recorded. When the product starts a new stage, the administrator inputs the required information into the system and tracks the progress by entering the percentage reported by the corresponding employee in charge of the product. Thus the schedules are tracked.

The last requirement is the manufacturer's timelines. It is the same like tracking the product's development schedule requiring completed percentage and due date, but it is related to the invoice created by the transaction between the company and the manufacturer. The table of manufacturer is needed to keep their information. Another table of outsourcing invoice between the employee in charge of the transaction and the manufacture is needed to keep the time of the invoice. In the invoice there can be many products which need to be outsourced so a new table of outsourced product is created between the invoice and the product. This table records the information discussed at the beginning, it also records the quantities of each product inside each invoice. Because each product may have different due date, even if the invoice is the same, the due date can be different. The timelines are thus kept.



## Major entities

Tables in relational database model require being in first normal form which means no attribute has multiple values in every table. The entities listed in the following thus have been considered to have only one value for each attribute. The following lists the major entities according to the above discussion in the relational database model:



### Supplier

Supplier is the game company which holds the IPR of the game and sells the game's license to the Acme Game Company. The Acme Company buys the licenses and sells the games to the customers. For convenience the first supplier is Acme Company itself because it reduces the possibilities of assigning null values when the IPR belongs to Acme Company. The license can also expire. The attributes of this entity are:

* Supplier ID: The identifying number of the supplier which is unique in this table for identifying different suppliers. It is a candidate primary key of this table;
* Supplier Name: The name of the supplier;
* Supplier Address: The address of the supplier showing its block number and street name;
* Supplier Contact: The contact number of the supplier;
* Suburb: Suburb name of the supplier's address;
* City: City name of the supplier's address;
* State: State name of the supplier's address;
* Postcode: Postcode of the supplier's address;
* Country: Country of the supplier.

### Age group

Age group is the group within which people's age varies from minimum age to maximum age. It is used to clarify the object or players of the game whether they are children or adults. It is separated as a new table because a game can be aimed at more than one age group. According to the definition of first normal form it can't be an attribute of the product. Its attributes are:

* Age group ID: The identifying number which is unique in this table. Every ID identifies different age groups;
* Minimum age: The lowest age of the age group such as 9 years old;
* Maximum age: The highest age of the age group such as 18 years old;
* Group Description: A description describing how people in this age group are like which is useful in marketing.

### Gender

Gender is detached as a new entity too because of the same reason of the Age Group. A game can be aimed at both genders. Its attributes are:

* Gender ID: An identifying number for both genders;
* Gender Name: Name of the gender such as "male" or "female".

### Distribution

It is the warehouse where the products are stored. It can be in different suburbs scattering in different cities. It is assumed the distributions are all in Australia. Its attributes are:

* Distribution ID: Identifying the warehouse in different places;
* Distribution Name: The name of the distribution which can be the same as the suburb;
* Suburb
* City
* State
* Postcode

### Inventory

It is a main entity of this database which is an alias of the product that is ready to be sold in stock. It contains two kinds of products: the licensed games and those designed by Acme Company. Both licenses will be called IPR but each product has an attribute of IPR type which can distinguish what kind of IPR it has. Its attributes are:

* Product ID: An identifying number of the product;
* Product International Standard Book Number (ISBN) : A serial number of the game identifying itself uniquely in this world;
* Product Name: The genuine name of the game excluding aliases;
* Product Genre: The category of the game such as "Action" or "Shoot";
* Product Description: Describing how the game is like;
* Product Price: The wholesale price of the game with which the retailer buys from Acme company;
* Product Quantity: The quantity of the game in stock;
* IPR ID: A unique number identifying the IPRs this database stores. Each game has only one IPR ID;
* IPR Type: The type of the IPR. Possible values are "License", "Trademark", "Patent" and "Industrial design right" etc. For "License" it means the IPR belongs to the other suppliers. For other types it means Acme Company holds the IPR;
* IPR Expiry Date: The date when the IPR expires. If a license expires, the administrator can notify the management to continue pay for it; if an IPR expires the company can renew it;
* Retail Price: Price with which the retailer should sell the product to the players. It is only a suggested price given by the retailer used for estimating the turnover of the retailer for the product;
* Supplier ID: The ID of the supplier indicating who holds the IPR of the game;
* Distribution ID: The ID of the distribution warehouse indicating where the product is stored.

### Game played

It is the associative entity derived by the many-to-many relationships among Inventory, Gender and Age Group. The name is given arbitrarily but it indicates only when the three factors which identify the product, gender and age group are present a specific game appetite can be determined. It can use a composite key of the three primary keys, but it is better to use a single key to replace the composite key for the sake of eliminating the following redundancy for second normal form. Thus its attributes determined are:

* Game ID: Identifying each instance of this table;
* Product ID: Foreign key from Inventory;
* Age Group ID: Foreign key from Age Group;
* Gender ID: Foreign key from Gender.

### Customer

Customers are retailers who buy the products from Acme Company with the wholesale prices. It is assumed that the customers are from different countries and they have a unique business number with each of them. This table stores related information about the customer. Its attributes are:

* Customer Name: Registered name of the retailer;
* Business Number: Retailer's business number;
* Customer Contact: Retailer's contact number;
* Customer Address: Retailer's registered address up to street name;
* Suburb
* City
* State
* Postcode
* Country

### Retail outlet

Retail outlets are the outlets belonging to the customers who are used to sell the products from either the Acme Company or others. It is assumed the retailer outlet sells only the specific customer's products such as the Coles ltd. and Dick Smith ltd. The storage of the information of the customers' outlets is for the convenience of distribution and contact. Its attributes are:

* Outlet ID: A number identifying the outlets inside this database;
* Outlet Address: The address of the outlet up to street name;
* Suburb
* City
* State
* Postcode
* Customer ID: Foreign key from the customer to specify whose outlet this is.

### Employee

The Employee table stores the information of the employees of Acme Company. These employees are responsible for different parts and they belong to different departments with different position titles. It is assumed they all live in Australia. Its attributes are:

* Employee Name: Name of the employee including surname and given name;
* Employee Age: Age of the employee;
* Employee Address
* Suburb
* City
* State
* Postcode
* Position Title: The title of the employee's position such as "Designer";
* Department: The department the employee is in such as "Marketing".

### Sales invoice

The Sales Invoice table records the invoices generated when the customer places an order. The time and customer is recorded in the invoice for tracking. Its attributes are:

* Invoice ID: The ID number identifying the invoice from others;
* Invoice Time: The time when the invoice is generated;
* Invoice Date: The date of the time;
* Invoice Month: The month of the time;
* Invoice Year: The year of the time;
* Customer ID: Foreign key from Customer table indicating which customer placed the order;
* Employee ID: Foreign key from Employee table indicating which employee is responsible for the transaction.

### Sales item

It is an associative entity derived between the Inventory table and the Sales Invoice table because of their many-to-many relationship. It uses the composite key of Product ID and Invoice ID to uniquely identify a sales item on different invoices. Because of the same reason as discussed in Game Played table, another primary key Item ID is used here instead of the composite key. The wholesale price isn't listed here because it is assumed the wholesale price is non-negotiable. Its attributes are:

* Item ID: An identifying number for different sales items here;
* Item Quantity: The quantity at which the customer buys the item;
* Product ID: Foreign key from the Inventory table to identify which product is this meaning;
* Invoice ID: Foreign key from Sales Invoice table indicating the invoice this item is on.

### Manufacture

This table stores the information about the manufacturers. These are mostly oversea factory owners who produce the games by the order of Acme Company. Its attributes are:

* Manufacture ID: The identifying number of the manufacturer in the database;
* Manufacture Name: The name of the manufacturer;
* Manufacture Contact: The contact number of the manufacturer;
* Manufacture Address
* Suburb
* City
* State
* Country
* Postcode

### Outsourcing invoice

This table keeps the records of the invoices when the transactions to the manufacturers are made successfully. It tracks the time, employee in charge and the manufacturer. Its attributes are:

* Invoice ID: The number used to distinguish different invoices;
* Invoice Time: The time when the invoice was generated;
* Invoice Date
* Invoice Month
* Invoice Year
* Employee ID: Foreign key from Employee table indicating the employee in charge of this invoice;
* Manufacture ID: The manufacturer who received the order.

### Production

This table stores the information of the products that are not yet fully developed by Acme Company. The products may be in any one of the four stages. For those after stage three may also have been protected by an IPR so information about the IPR such as IPR ID, expiry date must also be stored. It also stores the schedule of the development of the product such as completed percentage, start date and planned days. Its attributes are:

* Product ID: The identifying number distinguishing one another product in the database;
* Product Name: The temporary name of the product;
* Product Description: Description of the product;
* Product Genre: Category of the product;
* Product Stage: one of the four stages the product may be in;
* Completed percentage: The percentage completed for the product to get through its current stage;
* Start Date: The date when the product entered current stage;
* Planned Days: Number of days estimated for the product to get through current stage;
* IPR ID: The ID of the IPR stored in the database;
* IPR Expiry Date: Expiry date of the IPR;
* Employee ID: Indicating which employee is in charge of the product when it is in current stage. It can be null when the product is in third or fourth stage because it is indicated in some other tables too avoiding possible redundancy.

### Promotion event

This table stores the promotional events that are going to happen and those that have happened. It is useful for the management level to estimate if the company is successful in the promotion campaigns. It tracks the date, media type, duration and cost of the event. Its attributes are:

* Event ID: The ID of the event;
* Event Date: The date of the event;
* Event Media: The type of the media the event happens on;
* Promotion Cost: The cost for this event;
* Duration: The number of days this event will last for;
* Employee ID: The employee that is in charge of the event;
* Product ID: The product that is used in the specific event. It is assumed only one product is promoted every time.

### Outsourced product

It stores the products that appear on the invoices for outsourcing. It tracks the due date of the specific product, completed percentage reported from the manufacturer and quantity of the product in that invoice. It is also an associative entity between the Production table and the Invoice table and is identified by the product ID and the invoice ID. For the same reason discussed above, it uses another single primary key for both simplicity and avoiding redundancy. The attributes are:

* Item ID: An artificial ID number identifying items on the invoices;
* Outsourcing Cost: Specific cost for each item on each invoice because it may be different even for the same product and the same manufacturer;
* Quantity: The quantity for the specific item required by Acme Company;
* Due Date: The specific date for the item to be completed before;
* Completed Percentage: The current percentage of completion for the specific item reported by the manufacturer;
* Product ID: Foreign key indicating which product is this item;
* Invoice ID: Foreign key indicating which invoice is this item on.

# Task2: Conceptual Database Model



## Entity Relationship diagram in first normal form

The E-R diagram is like following:



Figure – E-R diagram in first normal form

## Relationships explanation

The relationships are explained in the following:

A supplier supplies no product meaning the supplier is only on the list but recently doesn't supply any product yet; or it can supply many products. A product can be supplied only by one supplier, and must have at least one supplier. The special supplier Acme Company is on top of the list.

A distribution has on supply of that product meaning it may be stored in other distribution; or it can store many products in one place. The product is stored in one place and cannot be stored in other places for assumption.

The relationship between Inventory, Gender and AgeGroup is ternary many-to-many relationship so the associative entity GamePlayed is created to represent this relationship. Generally a product can be not yet been played, an age group has not yet tried it or a kind of gender hasn't played it yet. The null cardinality on the many side is only logical and unimportant in the view of the database. What's important is the three non-key attributes in the GamePlayed table must be not null because an instance of the table must have the values from the three tables for the three attributes. A played game must be a product, have some in the age group and with one of the genders. Otherwise the product can't exist in this database.

The customer can have no retail outlet because it may sell the product online or it can have several outlets; a retail outlet belongs to only one customer and must have at least one customer as its owner.

The customer can place no order represented by the invoice or it can place many orders; an invoice belongs to only one customer and must have at least one customer responsible for it.

The associative entity SalesItem represents many-to-many relationship between the Inventory and the SaleInvoice entities. A product can be on many invoices with different items on each of it or no invoice sitting quietly in the distribution warehouse; the item must be one and only one of the products; An invoice must have at least one item or it is canceled and can have many items; the item must be on one and only one invoice.

An employee manages no invoice because he/she isn't in the sales department or many invoices; an invoice is managed by one employee and he/she should be the only one responsible for that transaction. This is the same for the promotion events and the outsourcing invoices. It is almost the same for the production but the product in development can have no employee in charge because when it is in stage three or four the employee in charge is also the same one in charge of the promotion event or the outsourcing transaction. At these stages the employee should be left blank as null for the product to avoid possible conflict and redundancy.

A product can have no promotion event because it is not in stage three or many events; a promotion event is for only one product and at least one product in assumption. The relationship between production and outsourcing invoices is many-to-many so an associative entity OutsourcedProduct is created to represent this relationship and the timeline of the manufacture. A product in stages other than four has no outsourced product but it can have many outsourced instance on different invoices because it can be outsourced for more than once.

Other relationships can be explained as those similar to above.

# Task3: Logical Database Model



## Entity Relationship diagram in third normal form

The E-R diagram is like following:



Figure – E-R diagram in third normal form

## Explanation of new entities and relationships

The primary keys of the tables are those appearing in the E-R diagram with "ID" suffix as the candidate key. The primary keys appearing in other tables as non-key attributes will be the foreign keys.

Because many tables need to specify the date, month and year in order to normalize the input of these data it is more convenient to create three separate tables for Date, Month and Year and assign them with fixed values so that when new table comes requiring the input of these three tables in can be done without violating referential integrity. The primary keys of these three tables use different IDs for simplicity and those tables referencing them will use foreign keys linking themselves to these three tables. By this way many tables appeared in first normal form will replace the attributes: date, month and year with DateID, MonthID and YearID.

This will be the same case for separating the address into four parts: country, state, city and suburb. Specific attributes can be added into these tables respectively such as acronyms and names. The Suburb table and the Country table don't use IDs because if using IDs with CountryCode and PostCode as attributes there could be functional dependency among the non-key values. Notice that every country code can decide the name of the country and every postcode determines the name of the suburb. To avoid this dependency and the redundancy thus generated the country code and the postcode will be used as primary key here directly. The address is now determined by the postcode linked from the suburb to the entity where suburb is linked to city, city to state and state to country. The chain of links makes the update of data orient only to that kind of data without affecting others making the process efficient. The drawback of this is data in parent tables needs to be as sufficient as possible in order to add data in child tables.

The relationship of Date to other entities is always mandatory one to optional many because in a day the entity appears optionally and many can appear but it can only appear in one day and only one day. The explanation is the same for Month, Year, Country, State, City and Suburb.

The Inventory table doesn't use Product ID as primary key because there is functional dependency for the ISBN and other attributes. Knowing the ISBN can uniquely determine any other attributes so it is necessary to either add a new table with ISBN as foreign key or replace the product ID with ISBN as the primary key.

For the same reason IPRType and IPR are separated as new tables. IPRType contains values such as "License", "Patent" and so on. The first entry is "License" for convenience of specifying those products provided by other suppliers. The foreign keys DateID, MonthID and YearID in IPR table indicate the time when the IPR expires. The supplierID is now linked to IPR instead of Inventory table because every product must have an IPR and tracking the supplier can be done by tracking its IPR. If Supplier is linked to the Inventory as before there may be functional dependency because the foreign key attribute IPRID in Inventory table can determine the supplier. The IPR table is also linked to the Production table so that all IPRs are managed together. An IPR can have no corresponding entry in Inventory table because it may be for those in Production table. But a product in Inventory table must have an IPR either it is license or other types. It is the same for Production table but it is optional for a production to have IPR because it may be still in design stage. For the one-to-one relationship between IPR and Inventory the foreign key is on the "null" side to avoid possible null value.

The product in development must also have attributes like age group and gender. It will be the same case as with Inventory and the relationship is many-to-many among them so a new associative entity GameTested is created to represent this relationship. Genre is detached as a new table linking both to Inventory and Production tables. A game must have one and only one genre but a genre can have no game in it or many games within it. The DateID, MonthID and YearID indicate the start date of current stage the product is in. The Stage is separated as a new table to maintain the stage information without the need of inserting new product record otherwise if no product is in that stage the information of it is lost.

For the same reason discussed above, the department and position are separated as new tables to maintain their information without the need of having the information of an employee of the specific department or position. An employee can have one and only one position and a position can have no employee or several employees. Many departments have same title name of positions such as "manager" so the position title can't determine any other attribute in Position table. Thus the position ID is used to uniquely identify a position in the company. It uses a foreign key DepartmentID to specify which department it belongs to. A department must have at least one position and a position must belong to one of the departments and can't belong to more than one department. The employee's department thus can be tracked by position he/she has. An employee can't be in more than one position in the company as in assumption. The manager of the employee can also be tracked as a foreign key in the non-key attributes referencing the primary key EmployeeID as a unary relationship. An employee can manage no one or many people and he/she can be managed by no one in the case of CEO or one direct manager. When querying the manager information of the employee the aliases for the same table Employee must be used to identify different instances of the table.

For the same reason of functional dependency a new table of CustomerBusiness is created because the business number can determine other attributes of the customer. A customer must have one and only one business number and a business number belongs to one and only one customer.

The date, month and year in OutsourcedProduct table indicate the due date of the manufacture of the product.

In order to maintain the information of media outlets that are available for promotion events, they are separated as a new table with attributes indicating their types and description. All promotion events will be in one and only one of the media types.

Note: The sequence numbers of the foreign keys in the E-R diagram are incorrect for some tables because of the order these tables were linked together but the number of them has been carefully inspected for each table.

## Data types and sizes

There is not enough space to display all tables in EER diagram in MySQL workbench. So the data types and sizes are listed as following alphabetically:

tblAgeGroup

* AgeGroupID: Integer
* MinimumAge: Int(2). All integers are stored in four bytes with size of -2147483648 to 2147483647 up to 11 digits. This is why telephone numbers can't use Integer as the data type. The "2" specifies only two digits are shown here but the storage space still requires four bytes for each value. (11.2.1 Integer Types (Exact Value) - INTEGER, INT, SMALLINT, TINYINT, MEDIUMINT, BIGINT, n. d.)
* MaximumAge: Int(2)
* GroupDescription: varchar(99). 99 characters can be shown here and if more characters are here they will be truncated. The administrator should remind the user of this in the user interface.

tblCity

* CityID: Integer
* CityName: varchar(19)
* StateID: Integer

tblCountry

* CountryCode: Integer
* CountryAcronym: varchar(9)
* CountryName: varchar(199)

tblCustomer

* CustomerID: Integer
* CustomerName: varchar(39)
* CustomerContact: varchar(19)
* CustomerAddress: varchar(39)
* PostCode: Integer

tblCustomerBusiness

* BusinessNumber: varchar(11)
* CustomerID: Integer

tblDate

* DateID: int(2)
* DateNumber: int(2)

tblDepartment

* DepartmentID: Integer
* DepartmentName: varchar(19)
* DepartmentDescription: varchar(99)

tblDistribution

* DistributionID: Integer
* DistributionName: varchar(29)
* DistributionAddress: varchar(39)
* DistributionContact: varchar(19)
* PostCode: Integer

tblEmployee

* EmployeeID: Integer
* EmployeeName: varchar(29)
* EmployeeAge: int(2)
* EmployeeAddress: varchar(29)
* PostCode: Integer
* ManagerID: Integer
* PositionID: Integer

tblGamePlayed

* GameID: Integer
* ProductISBN: varchar(29)
* AgeGroupID: Integer
* GenderID: Integer

tblGameTested

* GameID: Integer
* AgeGroupID: Integer
* GenderID: Integer
* ProductID: Integer

tblGender

* GenderID: int(1)
* GenderName: varchar(9)

tblGenre

* GenreID: Integer
* GenreName: varchar(19)
* GenreDescription: text. "Text" data type can display up to 65,535 characters. (SQL Data Types for Various DBs, n. d.)

tblInventory

* ProductISBN: varchar(29)
* ProductName: varchar(39)
* ProductDescription: varchar(299)
* ProductPrice: decimal(9,2). Total nine digits with two decimal places.
* ProductQuantity: Integer
* RetailPrice: decimal(9,2)
* LicenseFee:decimal(9,2)
* DistributionID: Integer
* GenreID: Integer
* IPRID: Integer

tblIPR

* IPRID: Integer
* YearID: Integer
* DateID: Integer
* MonthID: Integer
* CountryCode: Integer
* IPRTypeID: Integer
* SupplierID: Integer

tblIPRType

* IPRTypeID: Integer
* IPRType: varchar(99)

tblManufacture

* ManufactureID: Integer
* ManufactureName: varchar(39)
* ManufactureContact: varchar(19)
* ManufactureAddress: varchar(39)
* PostCode: Integer

tblMediaOutlet

* MediaOutletID: Integer
* MediaType: varchar(19)
* MediaDescription: varchar(99)

tblMonth

* MonthID: Integer
* MonthName: varchar(19)

tblOutsourcedProduct

* ItemID: Integer
* OutsourcingCost: decimal(9,2)
* Quantity: Integer
* CompletedPercentage: decimal(3,2)
* ProductID: Integer
* InvoiceID: Integer
* DateID: Integer
* MonthID: Integer
* YearID: Integer

tblOutsourcingInovice

* InvoiceID: Integer
* EmployeeID: Integer
* ManufactureID: Integer
* MonthID: Integer
* DateID: Integer
* YearID: Integer

tblPosition

* PositionID: Integer
* PositionTitle: varchar(19)
* PositionDescription: varchar(99)
* DepartmentID: Integer

tblProduction

* ProductID: Integer
* ProductName: varchar(29)
* ProductDescription: varchar(299)
* PlannedDays: Integer
* CompletedPercentage: decimal(3,2)
* EmployeeID: Integer
* YearID: Integer
* DateID: Integer
* MonthID: Integer
* GenreID: Integer
* IPRID: Integer
* StageID: Integer

tblPromotionEvent

* EventID: Integer
* PromotionCost: decimal(9,2)
* Duration: Integer
* EmployeeID: Integer
* ProductID: Integer
* YearID: Integer
* DateID: Integer
* MediaOutletID: Integer
* MonthID: Integer

tblRetailOutlet

* OutletID: Integer
* OutletName: varchar(39)
* OutletContact: varchar(19)
* OutletAddress: varchar(39)
* PostCode: Integer
* CustomerID: Integer

tblSalesInvoice

* InvoiceID: Integer
* InvoiceTime: time. The format of "Time" is "HH:MM:SS".
* CustomerID: Integer
* EmployeeID: Integer
* DateID: Integer
* MonthID: Integer
* YearID: Integer

tblSalesItem

* ItemID: Integer
* ItemQuantity: Integer
* ProductISBN: varchar(29)
* InvoiceID: Integer

tblStage

* StageID: Integer
* StageName: varchar(19)
* StageDescription: varchar(99)

tblState

* StateID: Integer
* StateName: varchar(29)
* StateAcronym: varchar(9)
* CountryCode: Integer

tblSuburb

* PostCode: Integer
* SuburbName: varchar(19)
* CityID: Integer

tblSupplier

* SupplierID: Integer
* SupplierName: varchar(29)
* SupplierAddress: varchar(29)
* SupplierContact: varchar(19)
* PostCode: Integer

tblYear

* YearID: Integer
* YearNumber: int(4)

# Task4: Database



## SQL codes for creating tables

The code is like following:

set foreign\_key\_checks=0;

create schema if not exists `project`;

use `project`;

#create parent tables first

create table if not exists project.tblIPRType(

`IPRTypeID` int not null auto\_increment,

`IPRType` varchar(99),

primary key(`IPRTypeID`),

unique index `IPRTypeID\_unique` (`IPRTypeID` asc)

)engine=InnoDB;

create table if not exists project.tblCountry(

`CountryCode` int not null,

`CountryAcronym` varchar(9),

`CountryName` varchar(199),

primary key(`CountryCode`),

unique index `CountryCode\_unique` (`CountryCode` asc)

)engine=InnoDB;

create table if not exists project.tblDate(

`DateID` int(2) not null auto\_increment,

`DateNumber` int(2),

primary key(`DateID`),

unique index `DateID\_unique`(`DateID` asc)

)engine=InnoDB;

create table if not exists project.tblMonth(

`MonthID` int(2) not null auto\_increment,

`MonthName` varchar(19),

primary key(`MonthID`),

unique index `MonthID\_unique` (`MonthID` asc)

)engine=InnoDB;

create table if not exists tblYear(

`YearID` int not null auto\_increment,

`YearNumber` int(4),

primary key(`YearID`),

unique index `YearID\_unique`(`YearID` asc)

)engine=InnoDB;

create table if not exists project.tblGenre(

`GenreID` int not null auto\_increment,

`GenreName` varchar(19),

`GenreDescription` text,

primary key(`GenreID`),

unique index `GenreID\_unique`(`GenreID` asc)

)engine=InnoDB;

create table if not exists project.tblAgeGroup(

`AgeGroupID` int not null auto\_increment,

`MinimumAge` int(2),

`MaximumAge` int(2),

`GroupDescription` varchar(99),

primary key(`AgeGroupID`),

unique index `AgeGroupID\_unique`(`AgeGroupID` asc)

)engine=InnoDB;

create table if not exists project.tblGender(

`GenderID` int(1) not null auto\_increment,

`GenderName` varchar(9),

primary key(`GenderID`),

unique index `GenderID\_unique`(`GenderID` asc)

)engine=InnoDB;

create table if not exists project.tblStage(

`StageID` int not null auto\_increment,

`StageName` varchar(19),

`StageDescription` varchar(99),

primary key(`StageID`),

unique index `StageID\_unique`(`StageID` asc)

)engine=InnoDB;

create table if not exists project.tblMediaOutlet(

`MediaOutletID` int not null auto\_increment,

`MediaType` varchar(19),

`MediaDescription` varchar(99),

primary key(`MediaOutletID`),

unique index `MediaOutletID\_unique`(`MediaOutletID` asc)

)engine=InnoDB;

create table if not exists project.tblDepartment(

`DepartmentID` int not null auto\_increment,

`DepartmentName` varchar(19),

`DepartmentDescription` varchar(99),

primary key(`DepartmentID`),

unique index `DepartmentID\_unique`(`DepartmentID` asc)

)engine=InnoDB;

#create child tables from those with least connections to the parent tables

create table if not exists project.tblPosition(

`PositionID` int not null auto\_increment,

`PositionTitle` varchar(19),

`PositionDescription` varchar(99),

`DepartmentID` int not null,

primary key(`PositionID`),

unique index `PositionID\_unique`(`PositionID` asc),

index `DepartmentToDetails\_idx`(`DepartmentID` asc),

constraint `DepartmentToDetails`

foreign key(`DepartmentID`)

references project.tblDepartment(`DepartmentID`)

on delete restrict

on update cascade

)engine=InnoDB;

create table if not exists project.tblState(

`StateID` int not null auto\_increment,

`StateName` varchar(29),

`StateAcronym` varchar(9),

`CountryCode` int not null,

primary key(`StateID`),

unique index `StateID\_unique`(`StateID` asc),

index `CountryToDetails\_idx`(`CountryCode` asc),

constraint `CountryToDetails`

foreign key(`CountryCode`)

references project.tblCountry(`CountryCode`)

on delete restrict

on update cascade

)engine=InnoDB;

create table if not exists project.tblCity(

`CityID` int not null auto\_increment,

`CityName` varchar(19),

`StateID` int not null,

primary key(`CityID`),

unique index `CityID\_unique`(`CityID` asc),

index `StateToDetails\_idx`(`StateID` asc),

constraint `StateToDetails`

foreign key(`StateID`)

references project.tblState(`StateID`)

on delete restrict

on update cascade

)engine=InnoDB;

create table if not exists project.tblSuburb(

`PostCode` int not null,

`SuburbName` varchar(19),

`CityID` int not null,

primary key(`PostCode`),

unique index `PostCode\_unique`(`PostCode` asc),

index `CityToDetails\_idx`(`CityID` asc),

constraint `CityToDetails`

foreign key(`CityID`)

references project.tblCity(`CityID`)

on delete restrict

on update cascade

)engine=InnoDB;

create table if not exists project.tblEmployee(

`EmployeeID` int not null auto\_increment,

`EmployeeName` varchar(29),

`EmployeeAge` int(2),

`EmployeeAddress` varchar(29),

`PostCode` int not null,

`ManagerID` int,

`PositionID` int not null,

primary key(`EmployeeID`),

unique index `EmployeeID\_unique`(`EmployeeID` asc),

index `SuburbToDetails\_idx`(`PostCode` asc),

index `ManagerToDetails\_idx`(`ManagerID` asc),

index `PositionToDetails\_idx`(`PositionID` asc),

constraint `SuburbToDetails`

foreign key(`PostCode`)

references project.tblSuburb(`PostCode`)

on delete restrict

on update cascade,

constraint `ManagerToDetails`

foreign key(`ManagerID`)

references project.tblEmployee(`EmployeeID`)

on delete restrict

on update cascade,

constraint `PositionToDetails`

foreign key(`PositionID`)

references project.tblPosition(`PositionID`)

on delete restrict

on update cascade

)engine=InnoDB;

create table if not exists project.tblSupplier(

`SupplierID` int not null auto\_increment,

`SupplierName` varchar(29),

`SupplierAddress` varchar(29),

`SupplierContact` varchar(19),

`PostCode` int not null,

primary key(`SupplierID`),

unique index `SupplierID\_unique`(`SupplierID` asc),

index `SupplierSuburbToDetails\_idx`(`PostCode` asc),

constraint `SupplierSuburbToDetails`

#constraint's name must be unique for each table inside the database

foreign key(`PostCode`)

references project.tblSuburb(`PostCode`)

on delete restrict

on update cascade

)engine=InnoDB;

create table if not exists project.tblDistribution(

`DistributionID` int not null auto\_increment,

`DistributionName` varchar(29),

`DistributionAddress` varchar(39),

`DistributionContact` varchar(19),

`PostCode` int not null,

primary key(`DistributionID`),

unique index `DistributionID\_unique`(`DistributionID` asc),

index `DistributionSuburbToDetails\_idx`(`PostCode` asc),

constraint `DistributionSuburbToDetails`

foreign key(`PostCode`)

references project.tblSuburb(`PostCode`)

on delete restrict

on update cascade

)engine=InnoDB;

create table if not exists project.tblCustomer(

`CustomerID` int not null auto\_increment,

`CustomerName` varchar(39),

`CustomerContact` varchar(19),

`CustomerAddress` varchar(39),

`PostCode` int not null,

primary key(`CustomerID`),

unique index `CustomerID\_unique`(`CustomerID` asc),

index `CustomerSuburbToDetails\_idx`(`PostCode` asc),

constraint `CustomerSuburbToDetails`

foreign key(`PostCode`)

references project.tblSuburb(`PostCode`)

on delete restrict

on update cascade

)engine=InnoDB;

create table if not exists project.tblCustomerBusiness(

`BusinessNumber` varchar(11) not null,

# "11" is the true length of the Australia Business Number and because the maximum number of integer is a 10-digit number "int" type can't be used here otherwise it will cause the "Duplicate entry for key" problem. (jp, 2009)

`CustomerID` int not null,

primary key(`BusinessNumber`),

unique index `BusinessNumber\_unique`(`BusinessNumber` asc),

index `CustomerToDetails\_idx`(`CustomerID` asc),

constraint `CustomerToDetails`

foreign key(`CustomerID`)

references project.tblCustomer(`CustomerID`)

on delete restrict

on update cascade

)engine=InnoDB;

create table if not exists project.tblManufacture(

`ManufactureID` int not null auto\_increment,

`ManufactureName` varchar(39),

`ManufactureContact` varchar(19),

`ManufactureAddress` varchar(39),

`PostCode` int not null,

primary key(`ManufactureID`),

unique index `ManufactureID\_unique`(`ManufactureID` asc),

index `ManufactureSuburbToDetails\_idx`(`PostCode` asc),

constraint `ManufactureSuburbToDetails`

foreign key(`PostCode`)

references project.tblSuburb(`PostCode`)

on delete restrict

on update cascade

)engine=InnoDB;

create table if not exists project.tblRetailOutlet(

`OutletID` int not null auto\_increment,

`OutletName` varchar(39),

`OutletContact` varchar(19),

`OutletAddress` varchar(39),

`PostCode` int not null,

`CustomerID` int not null,

primary key(`OutletID`),

unique index `OutletID\_unique`(`OutletID` asc),

index `RetailOutletSuburbToDetails\_idx`(`PostCode` asc),

index `RetailOutletCustomerToDetails\_idx`(`CustomerID` asc),

constraint `RetailOutletSuburbToDetails`

foreign key(`PostCode`)

references project.tblSuburb(`PostCode`)

on delete restrict

on update cascade,

constraint `RetailOutletCustomerToDetails`

foreign key(`CustomerID`)

references project.tblCustomer(`CustomerID`)

on delete restrict

on update cascade

)engine=InnoDB;

create table if not exists project.tblSalesInvoice(

`InvoiceID` int not null auto\_increment,

`InvoiceTime` time,

`CustomerID` int not null,

`EmployeeID` int not null,

`DateID` int not null,

`MonthID` int not null,

`YearID` int not null,

primary key(`InvoiceID`),

unique index `InvoiceID\_unique`(`InvoiceID` asc),

index `SalesInvoiceCustomerToDetails\_idx`(`CustomerID` asc),

index `SalesInvoiceEmployeeToDetails\_idx`(`EmployeeID` asc),

index `SalesInvoiceDateToDetails\_idx`(`DateID` asc),

index `SalesInvoiceMonthToDetails\_idx`(`MonthID` asc),

index `SalesInvoiceYearToDetails\_idx`(`YearID` asc),

constraint `SalesInvoiceCustomerToDetails`

foreign key(`CustomerID`)

references project.tblCustomer(`CustomerID`)

on delete restrict

on update cascade,

constraint `SalesInvoiceEmployeeToDetails`

foreign key(`EmployeeID`)

references project.tblEmployee(`EmployeeID`)

on delete restrict

on update cascade,

constraint `SalesInvoiceDateToDetails`

foreign key(`DateID`)

references project.tblDate(`DateID`)

on delete restrict

on update cascade,

constraint `SalesInvoiceMonthToDetails`

foreign key(`MonthID`)

references project.tblMonth(`MonthID`)

on delete restrict

on update cascade,

constraint `SalesInvoiceYearToDetails`

foreign key(`YearID`)

references project.tblYear(`YearID`)

on delete restrict

on update cascade

)engine=InnoDB;

create table if not exists project.tblIPR(

`IPRID` int not null auto\_increment,

`YearID` int not null,

`DateID` int not null,

`MonthID` int not null,

`CountryCode` int not null,

`IPRTypeID` int not null,

`SupplierID` int not null,

primary key(`IPRID`),

unique index `IPRID\_unique`(`IPRID` asc),

index `IPRYearToDetails\_idx`(`YearID` asc),

index `IPRDateToDetails\_idx`(`DateID` asc),

index `IPRMonthToDetails\_idx`(`MonthID` asc),

index `IPRCountryToDetails\_idx`(`CountryCode` asc),

index `IPRTypeToDetails\_idx`(`IPRTypeID` asc),

index `SupplierToDetails\_idx`(`SupplierID` asc),

constraint `IPRYearToDetails`

foreign key(`YearID`)

references project.tblYear(`YearID`)

on delete restrict

on update cascade,

constraint `IPRDateToDetails`

foreign key(`DateID`)

references project.tblDate(`DateID`)

on delete restrict

on update cascade,

constraint `IPRMonthToDetails`

foreign key(`MonthID`)

references project.tblMonth(`MonthID`)

on delete restrict

on update cascade,

constraint `IPRCountryToDetails`

foreign key(`CountryCode`)

references project.tblCountry(`CountryCode`)

on delete restrict

on update cascade,

constraint `IPRTypeToDetails`

foreign key(`IPRTypeID`)

references project.tblIPRType(`IPRTypeID`)

on delete restrict

on update cascade,

constraint `SupplierToDetails`

foreign key(`SupplierID`)

references project.tblSupplier(`SupplierID`)

on delete restrict

on update cascade

)engine=InnoDB;

create table if not exists project.tblInventory(

`ProductISBN` varchar(29) not null,

`ProductName` varchar(39),

`ProductDescription` varchar(299),

`ProductPrice` decimal(9,2),

#maximum number allowed is 9,999,999.99

`ProductQuantity` int,

`RetailPrice` decimal(9,2),

`LicenseFee` decimal(9,2),

`DistributionID` int not null,

`GenreID` int not null,

`IPRID` int not null,

primary key(`ProductISBN`),

unique index `ProductISBN\_unique`(`ProductISBN` asc),

index `DistributionToDetails\_idx`(`DistributionID` asc),

index `GenreToDetails\_idx`(`GenreID` asc),

index `IPRToDetails\_idx`(`IPRID` asc),

constraint `DistributionToDetails`

foreign key(`DistributionID`)

references project.tblDistribution(`DistributionID`)

on delete restrict

on update cascade,

constraint `GenreToDetails`

foreign key(`GenreID`)

references project.tblGenre(`GenreID`)

on delete restrict

on update cascade,

constraint `IPRToDetails`

foreign key(`IPRID`)

references project.tblIPR(`IPRID`)

on delete restrict

on update cascade

)engine=InnoDB;

create table if not exists project.tblSalesItem(

`ItemID` int not null auto\_increment,

`ItemQuantity` int,

`ProductISBN` varchar(29) not null,

`InvoiceID` int not null,

primary key(`ItemID`),

unique index `ItemID\_unique`(`ItemID` asc),

index `ProductToDetails\_idx`(`ProductISBN` asc),

index `InvoiceToDetails\_idx`(`InvoiceID` asc),

constraint `ProductToDetails`

foreign key(`ProductISBN`)

references project.tblInventory(`ProductISBN`)

on delete restrict

on update cascade,

constraint `InvoiceToDetails`

foreign key(`InvoiceID`)

references project.tblSalesInvoice(`InvoiceID`)

on delete restrict

on update cascade

)engine=InnoDB;

create table if not exists project.tblGamePlayed(

`GameID` int not null auto\_increment,

`ProductISBN` varchar(29) not null,

`AgeGroupID` int not null,

`GenderID` int not null,

primary key(`GameID`),

unique index `GameID\_unique`(`GameID`),

index `GamePlayedProductToDetails\_idx`(`ProductISBN` asc),

index `GamePlayedAgeGroupToDetails\_idx`(`AgeGroupID` asc),

index `GamePlayedGenderToDetails\_idx`(`GenderID` asc),

constraint `GamePlayedProductToDetails`

foreign key(`ProductISBN`)

references project.tblInventory(`ProductISBN`)

on delete restrict

on update cascade,

constraint `GamePlayedAgeGroupToDetails`

foreign key(`AgeGroupID`)

references project.tblAgeGroup(`AgeGroupID`)

on delete restrict

on update cascade,

constraint `GamePlayedGenderToDetails`

foreign key(`GenderID`)

references project.tblGender(`GenderID`)

on delete restrict

on update cascade

)engine=InnoDB;

create table if not exists project.tblOutsourcingInvoice(

`InvoiceID` int not null auto\_increment,

`EmployeeID` int not null,

`ManufactureID` int not null,

`MonthID` int not null,

`DateID` int not null,

`YearID` int not null,

primary key(`InvoiceID`),

unique index `InvoiceID\_unique`(`InvoiceID` asc),

index `EmployeeToDetails\_idx`(`EmployeeID` asc),

index `ManufactureToDetails\_idx`(`ManufactureID` asc),

index `MonthToDetails\_idx`(`MonthID` asc),

index `DateToDetails\_idx`(`DateID` asc),

index `YearToDetails\_idx`(`YearID` asc),

constraint `EmployeeToDetails`

foreign key(`EmployeeID`)

references project.tblEmployee(`EmployeeID`)

on delete restrict

on update cascade,

constraint `ManufactureToDetails`

foreign key(`ManufactureID`)

references project.tblManufacture(`ManufactureID`)

on delete restrict

on update cascade,

constraint `MonthToDetails`

foreign key(`MonthID`)

references project.tblMonth(`MonthID`)

on delete restrict

on update cascade,

constraint `DateToDetails`

foreign key(`DateID`)

references project.tblDate(`DateID`)

on delete restrict

on update cascade,

constraint `YearToDetails`

foreign key(`YearID`)

references project.tblYear(`YearID`)

on delete restrict

on update cascade

)engine=InnoDB;

create table if not exists project.tblProduction(

`ProductID` int not null auto\_increment,

`ProductName` varchar(29),

`ProductDescription` varchar(299),

`CompletedPercentage` decimal(3,2),

#maximum value is "9.99" because the project can be completed before deadline

`PlannedDays` int,

`EmployeeID` int,

`YearID` int not null,

`DateID` int not null,

`MonthID` int not null,

`GenreID` int not null,

`IPRID` int,

`StageID` int not null,

primary key(`ProductID`),

unique index `ProductID\_unique`(`ProductID` asc),

index `ProductionEmployeeToDetails\_idx`(`EmployeeID` asc),

index `ProductionYearToDetails\_idx`(`YearID` asc),

index `ProductionDateToDetails\_idx`(`DateID` asc),

index `ProductionMonthToDetails\_idx`(`MonthID` asc),

index `ProductionGenreToDetails\_idx`(`GenreID` asc),

index `ProductionIPRToDetails\_idx`(`IPRID` asc),

index `StageToDetails\_idx`(`StageID` asc),

constraint `ProductionEmployeeToDetails`

foreign key(`EmployeeID`)

references project.tblEmployee(`EmployeeID`)

on delete restrict

on update cascade,

constraint `ProductionYearToDetails`

foreign key(`YearID`)

references project.tblYear(`YearID`)

on delete restrict

on update cascade,

constraint `ProductionDateToDetails`

foreign key(`DateID`)

references project.tblDate(`DateID`)

on delete restrict

on update cascade,

constraint `ProductionMonthToDetails`

foreign key(`MonthID`)

references project.tblMonth(`MonthID`)

on delete restrict

on update cascade,

constraint `ProductionGenreToDetails`

foreign key(`GenreID`)

references project.tblGenre(`GenreID`)

on delete restrict

on update cascade,

constraint `ProductionIPRToDetails`

foreign key(`IPRID`)

references project.tblIPR(`IPRID`)

on delete restrict

on update cascade,

constraint `StageToDetails`

foreign key(`StageID`)

references project.tblStage(`StageID`)

on delete restrict

on update cascade

)engine=InnoDB;

create table if not exists project.tblPromotionEvent(

`EventID` int not null auto\_increment,

`PromotionCost` decimal(9,2),

`Duration` int,

#Duration is the number of days for promotion

`EmployeeID` int not null,

`ProductID` int not null,

`YearID` int not null,

`DateID` int not null,

`MediaOutletID` int not null,

`MonthID` int not null,

primary key(`EventID`),

unique index `EventID\_unique`(`EventID` asc),

index `EventEmployeeToDetails\_idx`(`EmployeeID` asc),

index `EventProductToDetails\_idx`(`ProductID` asc),

index `EventYearToDetails\_idx`(`YearID` asc),

index `EventDateToDetails\_idx`(`DateID` asc),

index `EventMediaOutletToDetails\_idx`(`MediaOutletID` asc),

index `EventMonthToDetails\_idx`(`MonthID` asc),

constraint `EventEmployeeToDetails`

foreign key(`EmployeeID`)

references project.tblEmployee(`EmployeeID`)

on delete restrict

on update cascade,

#if the employee resigned, the project can be passed onto someone else before the employee's record is deleted and only those who are not referenced can be deleted

constraint `EventProductToDetails`

foreign key(`ProductID`)

references project.tblProduction(`ProductID`)

on delete restrict

on update cascade,

constraint `EventYearToDetails`

foreign key(`YearID`)

references project.tblYear(`YearID`)

on delete restrict

on update cascade,

constraint `EventDateToDetails`

foreign key(`DateID`)

references project.tblDate(`DateID`)

on delete restrict

on update cascade,

constraint `EventMediaOutletToDetails`

foreign key(`MediaOutletID`)

references project.tblMediaOutlet(`MediaOutletID`)

on delete restrict

on update cascade,

constraint `EventMonthToDetails`

foreign key(`MonthID`)

references project.tblMonth(`MonthID`)

on delete restrict

on update cascade

)engine=InnoDB;

create table if not exists project.tblOutsourcedProduct(

`ItemID` int not null auto\_increment,

`OutsourcingCost` decimal(9,2),

`Quantity` int,

`CompletedPercentage` decimal(3,2),

`ProductID` int not null,

`InvoiceID` int not null,

`DateID` int not null,

`MonthID` int not null,

`YearID` int not null,

primary key(`ItemID`),

unique index `ItemID\_unique`(`ItemID` asc),

index `OutsourcedProductToDetails\_idx`(`ProductID` asc),

index `OutsourcedInvoiceToDetails\_idx`(`InvoiceID` asc),

index `OutsourcedDateToDetails\_idx`(`DateID` asc),

index `OutsourcedMonthToDetails\_idx`(`MonthID` asc),

index `OutsourcedYearToDetails\_idx`(`YearID` asc),

constraint `OutsourcedProductToDetails`

foreign key(`ProductID`)

references project.tblProduction(`ProductID`)

on delete restrict

on update cascade,

constraint `OutsourcedInvoiceToDetails`

foreign key(`InvoiceID`)

references project.tblOutsourcingInvoice(`InvoiceID`)

on delete restrict

on update cascade,

constraint `OutsourcedDateToDetails`

foreign key(`DateID`)

references project.tblDate(`DateID`)

on delete restrict

on update cascade,

constraint `OutsourcedMonthToDetails`

foreign key(`MonthID`)

references project.tblMonth(`MonthID`)

on delete restrict

on update cascade,

constraint `OutsourcedYearToDetails`

foreign key(`YearID`)

references project.tblYear(`YearID`)

on delete restrict

on update cascade

)engine=InnoDB;

create table if not exists project.tblGameTested(

`GameID` int not null auto\_increment,

`AgeGroupID` int,

`GenderID` int,

`ProductID` int,

primary key(`GameID`),

unique index `GameID\_unique`(`GameID` asc),

index `GameTestedAgeGroupToDetails\_idx`(`AgeGroupID` asc),

index `GameTestedGenderToDetails\_idx`(`GenderID` asc),

index `GameTestedProductToDetails\_idx`(`ProductID` asc),

constraint `GameTestedAgeGroupToDetails`

foreign key(`AgeGroupID`)

references project.tblAgeGroup(`AgeGroupID`)

on delete restrict

on update cascade,

constraint `GameTestedGenderToDetails`

foreign key(`GenderID`)

references project.tblGender(`GenderID`)

on delete restrict

on update cascade,

constraint `GameTestedProductToDetails`

foreign key(`ProductID`)

references project.tblProduction(`ProductID`)

on delete restrict

on update cascade

)engine=InnoDB;

## SQL codes for inserting sample records

The code is like following:

Set foreign\_key\_checks=1;

#enable foreign key check to avoid possible referential integrity violation

use project;

insert into tblIPRType(`IPRType`)

values('License'),

('Patent'),

('Copyright'),

('Trademarks'),

('Trade Dress'),

('Trade Secret'),

('Industrial Design Rights');

#(Intellectual property, n. d.)

insert into tblCountry()

values(86,'CN','China'),

(61,'AU','Australia'),

(1,'US','United States');

#(List of country calling codes, n. d.)

#the following uses a stored procedure and calls it for inserting records into the tblDate table automatically

delimiter $$

#redefine the default delimiter ";" to "$$" temporarily to pass the ";" inside the body function to the SQL server (Ahsan, 2010)

drop procedure if exists InsertDate$$

create procedure InsertDate()

#"()" is necessary

begin

#"begin" and "end" is a pair of keywords

declare i int;

#declare the variable

set i=1;

#initialize the value as 1

while i <=31 do

insert into tblDate(`DateNumber`)

values(i);

#the primary key column increments automatically

set i=i+1;

#loop continues until i > 31

end while;

end $$

#ending definition of routine using newly defined delimiter

delimiter ;

#redefine the delimiter to default

call InsertDate();

#call the routine to run

insert into tblMonth(`MonthName`)

values('January'),

('February'),

('March'),

('April'),

('May'),

('June'),

('July'),

('August'),

('September'),

('October'),

('November'),

('December');

#use the routine again to insert values into tblYear

delimiter $$

drop procedure if exists InserYear$$

create procedure InsertYear()

begin

declare i int;

set i=1998;

#company started from 1998

while i <=2020 do

insert into tblYear(`YearNumber`)

values(i);

set i=i+1;

end while;

end$$

delimiter ;

call InsertYear();

insert into tblGenre(`GenreName`,`GenreDescription`)

values('RPG',"Role Play Game lets the player play someone to venture and increase its level"),

('Action',"Video game lets the player play someone to venture featuring the actions in fight or overcoming obstacles"),

('Shoot',"Shoot game visualizes the real battle onto the screen and lets the player to shoot by pointing its cursor"),

('Tactical',"Tactical game let the player play the role of a commander to deploy its army or construct its power in order to win");

insert into tblAgeGroup(`MinimumAge`,`MaximumAge`,`GroupDescription`)

values(9,11,'Junior middle school students'),

(12,18,'The teenagers');

insert into tblGender(`GenderName`)

values('Male'),

('Female');

insert into tblStage(`StageName`,`StageDescription`)

values('First Stage','Design team develops a new game'),

('Second Stage','Program team writes and tests the code on different platforms'),

('Third Stage','Sales and marketing team promote the product via media outlets'),

('Fourth Stage','Production team outsources the product for mass production');

insert into tblMediaOutlet(`MediaType`,`MediaDescription`)

values('TV','Advertise the game on TV program or advertisement'),

('Internet','Introduce the game on websites'),

('Billboard','Advertise the game on billboards alongside the streets');

insert into tblDepartment(`DepartmentName`,`DepartmentDescription`)

values('Human Resource','Managing matters concerning the employees'),

('Sales','Responsible for products sold to retailers'),

('Marketing','Responsible for advertising the product'),

('Design','Responsible for designing new games'),

('Program','Responsible for coding and testing'),

('Production','Responsible for designing and producing the game');

insert into tblPosition(`PositionTitle`,`PositionDescription`,`DepartmentID`)

values('HR Manager','GM of HR',1),

('Salesperson','Sales representative',2),

('Sales Manager','Manager of sales',2),

('Host','Marketing host',3),

('Marketing Manager','Manager of marketing',3),

('Designer','Designs the game',4),

('Programmer','Codes and tests the game',4),

('Outsourcer','Outsources the game',4),

('Production Manager','Manager of production',4);

insert into tblState(`StateName`,`StateAcronym`,`CountryCode`)

values('West Australia','WA',61),

('New South Wales','NSW',61),

('Guangdong Province','GD',86),

('Macao S.A.R.','MO',86),

('Califonia','CA',1),

('HongKong S.A.R.','HK',86);

insert into tblCity(`CityName`,`StateID`)

values('Perth',1),

('Sydney',2),

('Melbourne',2),

('Guangzhou',3),

('Shunde',3),

('Macao',4),

('Los Angles',5),

('Hongkong',6);

insert into tblSuburb()

values(6060,'Yokine',1),

(6054,'Maddington',1),

(2176,'Abbotsbury',2),

(2093,'Balgowlah',2),

(3196,'Edithvale',3),

(3194,'Mentone',3),

(511400,'Panyu precinct',4),

(510000,'Haizhu precinct',4),

(528329,"Jun'an town",5),

(528311,'Beijiao town',5),

(999078,'Macao',6),

(91331,'Arleta',7),

(999077,'Hongkong',8);

Set foreign\_key\_checks=0;

#necessary because the Employee table references its primary key which may be inserted later

insert into tblEmployee(`EmployeeName`,`EmployeeAge`,`EmployeeAddress`,`PostCode`,`ManagerID`,`PositionID`)

values('Man Fu Lei',29,'301 Cape Street',6060,null,1),

('Isaac Khoza',23,'15 Burslem Drive',6054,3,2),

('Ugyne Dema',28,'34 Pingston Street',2176,5,4),

('Yurou Tang',19,'64 Xianjian Avenue',2093,9,6),

('Weiwei Zhang',30,'33 Tongling Road',3196,9,7),

('Yun Ma',46,'26 Aliyun Street',3194,9,8),

('Huateng Ma',55,'16 Qkill Road',6060,1,3),

('Hongwei Zhou',39,'360 Court Avenue',6054,1,5),

('Jacky Cheng',61,'16 House Street',2176,1,9);

Set foreign\_key\_checks=1;

insert into tblSupplier(`SupplierName`,`SupplierAddress`,`SupplierContact`,`PostCode`)

values('AcmeGame','19 Waneroo Road','37563456',6060),

#The first one is the in-house record if IPR isn't a license it must be chosen

('Ubisoft','17 Montreal Street','15344822671',528329),

('Blizzard','19 America Avenue','28455301',999078),

('Softstar','23 Beijing Road','13802689261',510000),

('Mamamia','999 Helpme Avenue','13709394',91331),

('Kingsoft','11 Jinshan Street','92283317',2093);

insert into tblDistribution

(`DistributionName`,`DistributionAddress`,`DistributionContact`,`PostCode`)

values('Warehouse-Perth','17 Flinders Street','92273212',6060),

('Warehouse-Sydney','19 Chedan Avenue','51677899',2176),

('Warehouse-Melbourne','20 Ono Road','13709333',3196);

insert into tblCustomer

(`CustomerName`,`CustomerContact`,`CustomerAddress`,`PostCode`)

values('Coles','93795013','17 Bumbum Avenue',2176),

('Woolwirth','87391471','29 Feihua Street',3194),

('Super L','98765431','155 Rolling Road',2093),

('The Good Guys','12345678','333 Beckam Street',6054),

('Dick Smith','23456234','44 Wiki Street',2093);

insert into tblCustomerBusiness()

values(12345678901,1),

('98765432109',2),

('34537563857',3),

('37532789079',4),

('93749878978',5);

insert into tblManufacture

(`ManufactureName`,`ManufactureContact`,`ManufactureAddress`,`PostCode`)

values('Tianhe Game Ltd.','86756742','33 Dongdong Street',510000),

('Wanghan Hardware','485677959','23 Jiuming Avenue',511400),

('Dongsheng Moto Ltd.','1256857622','33 Gogo Road',528329),

('Aidele Gameware','6894677922','34 Shangli Village',528311);

insert into tblRetailOutlet

(`OutletName`,`OutletContact`,`OutletAddress`,`PostCode`,`CustomerID`)

values('Coles Dogswamp','98756789','11 Flinders Street',6060,1),

('Dakuaiho','34567865','22 Binghamton Avenue',3196,1),

('Nufaconguan','67586749','33 Pinlanch Street',6054,2),

('Xiaoyuxie','67857498','44 Taiwangyan Road',2093,3),

('Yangtian','64542718','55 Changxiao Street',2176,4),

('Zhuanghuai','77685432','67 Jily Avenue',3194,5);

insert into tblSalesInvoice

(`InvoiceTime`,`CustomerID`,`EmployeeID`,`DateID`,`MonthID`,`YearID`)

values('14:25:22',1,2,31,3,2),

('15:33:24',1,2,5,4,2),

('11:30:23',2,2,6,5,2),

('15:44:43',3,7,15,8,3),

('14:33:22',4,7,11,9,3),

('16:44:33',5,2,24,9,11),

('11:22:33',3,2,18,2,17),

('13:44:55',2,2,3,3,15);

insert into tblIPR

(`YearID`,`DateID`,`MonthID`,`CountryCode`,`IPRTypeID`,`SupplierID`)

values(23,31,12,86,1,3),

(22,31,9,61,2,1),

(21,1,1,1,3,1),

(22,1,1,61,1,5),

(21,31,9,86,7,1),

(22,4,4,86,3,1),

(20,9,9,1,1,4),

(21,11,11,61,2,1),

(22,29,12,1,1,2),

(23,10,9,61,4,1),

(22,12,12,86,2,1),

(21,11,12,61,7,1);

insert into tblInventory

(`ProductISBN`,`ProductName`,`ProductDescription`,`ProductPrice`,`ProductQuantity`,`RetailPrice`,`LicenseFee`,`DistributionID`,`GenreID`,`IPRID`)

values('349436-446-345345',"Assasin's creed",'Player acting as an assasin completes missions and kills enemies',12.10,111,44.99,56748,3,2,1),

('868956-345-473978','The Frozen Throne',"Player produces and controls its soldiers to defeat the enemy's army",3.20,222,15.99,null,2,4,2),

('939572-473-397543','Prince of Persia','Player acting as the prince to jump over obstacles and kill the enemies with the power of rewinding',11.10,333,19.99,null,1,2,3),

('359349-347-237494','Legend of Paladin V','Player controls the team of main actors to venture and gain experience by killing monsters',14.20,444,39.99,34579,2,1,4),

('320599-309-389475','Call of Duty','Player controls the cursor to shoot the enemies',18.20,333,38.99,null,3,3,5),

('349887-475-497500','Bio Hazard','Player controls the main actress to venture and shoot the zombies',13.20,222,29.99,null,2,3,6),

('394739-375-374993','Xuanyuan Sword III','Player controls the team of main actors to venture and experience the sensitive story of their legend',14.20,111,34.99,13479,1,1,7),

('390759-374-973498','Amazing Spiderman','Player controls the spiderman to defeat the criminals using his abilities',35.10,222,76.99,null,2,2,8),

('973495-934-359407','Diablo II','Player chooses the career and gains experience by killing monsters',14.10,333,34.99,4956,3,1,9),

('934754-984-397459','Age of Empire II','Player chooses a race and develops its own army until defeating others',13.10,222,29.99,null,2,4,10);

insert into tblSalesItem(`ItemQuantity`,`ProductISBN`,`InvoiceID`)

values(55,'934754-984-397459',1),

(46,'973495-934-359407',2),

(101,'349887-475-497500',3),

(49,'868956-345-473978',3),

(155,'394739-375-374993',4),

(222,'939572-473-397543',5),

(211,'359349-347-237494',5),

(177,'349887-475-497500',6),

(155,'390759-374-973498',7),

(78,'320599-309-389475',8),

(67,'934754-984-397459',8);

insert into tblGamePlayed

(`ProductISBN`,`AgeGroupID`,`GenderID`)

values('320599-309-389475',2,1),

('349436-446-345345',2,1),

('349887-475-497500',2,1),

('359349-347-237494',2,1),

('359349-347-237494',2,2),

('390759-374-973498',1,1),

('390759-374-973498',2,1),

('394739-375-374993',2,1),

('394739-375-374993',2,2),

('868956-345-473978',2,1),

('934754-984-397459',2,1),

('939572-473-397543',1,1),

('939572-473-397543',2,1),

('973495-934-359407',2,1),

('973495-934-359407',2,2);

insert into tblOutsourcingInvoice

(`EmployeeID`,`ManufactureID`,`MonthID`,`DateID`,`YearID`)

values(6,2,8,14,15),

(6,3,7,19,16);

insert into tblProduction

(`ProductName`,`ProductDescription`,`CompletedPercentage`,`PlannedDays`,`EmployeeID`,`YearID`,`DateID`,`MonthID`,`GenreID`,`IPRID`,`StageID`)

values('Tomb raider','An actress ventures to different ancient tombs to discover the treasuries',null,300,null,14,19,9,2,11,4),

('Virus company','A tactical game using strategies to propogate the virus to the world',null,108,null,15,23,7,4,12,4),

('Counter Strike','A shoot game where players play two teams to defeat each other',0.3,111,null,16,3,3,3,null,3),

('Plants VS zombies','A tactical lovely game using plants to defeat zombies',0.5,222,null,17,1,1,4,null,3),

('Devil May Cry','An action game controlling the main actor to kill monsters',0.7,94,5,17,9,9,2,null,2),

('Sanguo wushuang','an action game where player kills enemies in the battle field using skills',0.2,120,4,17,16,8,2,null,1);

insert into tblPromotionEvent

(`PromotionCost`,`Duration`,`EmployeeID`,`ProductID`,`YearID`,`DateID`,`MonthID`,`MediaOutletID`)

values(234589.35,30,3,3,16,15,5,1),

(34758.65,60,8,3,16,15,8,2),

(394759.45,120,3,4,17,3,2,3),

(89380.10,60,8,4,17,7,7,1);

insert into tblOutsourcedProduct

(`CompletedPercentage`,`OutsourcingCost`,`Quantity`,`ProductID`,`InvoiceID`,`DateID`,`MonthID`,`YearID`)

values(1.0,348975.35,300,1,1,14,9,16),

(1.0,239845.65,400,2,1,14,11,16),

(0.4,308432.25,500,1,2,19,5,18),

(0.6,394598.55,400,2,2,19,3,18);

insert into tblGameTested

(`ProductID`,`AgeGroupID`,`GenderID`)

values(1,1,1),

(1,2,1),

(2,2,1),

(2,2,2),

(3,2,1),

(4,1,1),

(4,1,2),

(4,2,1),

(4,2,2),

(5,2,1),

(6,1,1),

(6,2,1);

## Screenshots

The screenshots show the tables are created and sample data is inserted:

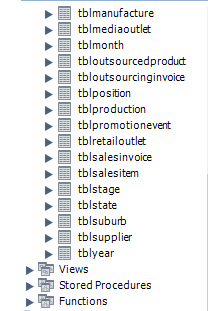
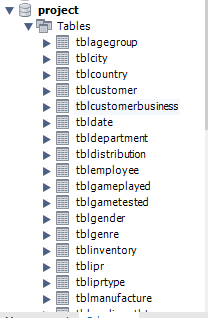


Figure – Created tables

Screenshots for insertion of sample data are available in Appendix.

# Task5: User Interface



## User interface design

## Forms

The database uses different kinds of forms to accept queries and update data. The query forms should consist of conditions for clients to query the database for useful information such as that specifying the start date and end date to query a detailed list of products sold. The query form is used when the manager needs to know some specific information so it is designed according to the possible business needs. Some pre-defined functions thus can be inserted as embedded SQL into the form as fixed functions such as finding the most successful retailer. The update form is different and it is only needed when certain event happens such as that a customer places an order and the related tables such as Invoice and Sales Item and so on need to be updated. These forms thus should be designed as accepting new records for related tables.

The following is a sample form for querying sales or inventory information:



Figure – Sample query form

The related information about a product can be input into the form to query its related information. None of the fields is mandatory. For example, to query the sales report within certain month and year the start date and end date can be specified while leaving others blank and click on the "Sales report" button. The sales report listing the products sold within the specified time with detailed information will display. If the user wants to know about the inventory about the price, he/she can input the lowest price and highest price to find the detailed list of products within this price range in the inventory. It also has a pre-defined function which displays which customer is the most successful. It saves the user's time and fits for the business need. Other queries can be done in similar way and clicking on different buttons generates different kinds of reports.

## Reports

The report is a response of the submitted form. It displays either the required information or a status report showing whether the update of information is successful. It should be designed in a clear and neat style where users can easily recognize useful information. It should also have the function of ordering the list by any column shown there so that users can compare the items from different aspect. An easy navigating system is also necessary for users to be clear of the structure of the report. For the sales report, the list of products with their attributes such as name, ISBN, age group, gender, prices and so on should be displayed for marketing department to analyze the trend of the market. For the inventory report the IPR information, supplier name, quantity in stock and distribution name should be displayed for administrators to check for products in shortage.

The following is a sample report as a response of the above form:



Figure – Sample report

In the report the titles are in capital letters for being outstanding. It has a summary area showing the summary of the report such as the number of products involved, age groups, gender and so on. It also displays the lowest price and highest price in the actual report and the start date as well as the end date entered by the user. The display format is word aligned left and number aligned right. It has a navigation system showing which page the user is navigating.

The little arrow right next to each column of the details panel can be clicked to choose ascending order by this column or descending or simply hide this column. For example, if a user wants to sort the list by the most profitable products from highest to lowest he/she can click on the arrow next to "TURNOVER" column and choose "descending order". The checkboxes in the "9-11 YEAR" and "12-18 YEAR" columns can show the many-to-many relationship clearly. Because relational database doesn't have multiple values for an attribute, the only way to display all useful information is to list all possible records in the table. The checkboxes can show this kind of information easily without the need of duplicating the records. The "QUANTITY" here means the quantity sold in the time range specified.

If it is an inventory report, the columns shown will be a little different such as that the distribution name will be displayed and the start date and end date will be gone because it is assumed they don't have business meaning in this database model.

## Database functions

The database is designed based on the thin client mode in three-tier architecture. The thin client mode means the client doesn't need any additional application to query the database and it only needs a browser to present the webpage. This is the first tier. In the second tier there are two parts: web server and application server. The clients send the normal webpage to the web server which determines if the page needs to invoke query to the database server. If it does, the web server will invoke the platforms such as Java Servlet Pages (JSP) to handle the correct type of query forms. It will then register the drivers and open the connection to the database server which responds with correct data and passes control to the JSP. (Hoffer, Ramesh& Topi, 2011) The database server is the third tier which stores the procedures and the database itself to create a kind of thin client and fat server.

The update queries are done when certain events happen such as when the clients place an order, a product starts to be outsourced and so on. Other kinds of queries are used when the manager needs to know about certain information of the products such as sales and inventory report. The concerns the author described in the requirement analysis section can be solved here:

* Inventory control: Certain queries such as product quantities in stock can be run regularly such as every week to check if there is product that needs to be filled. If occasionally a product needs to be transferred from one distribution to another, the information can be updated by the update query.
* Sales and marketing updates: When a customer places an order or updates an existing order, involved tables can be updated by inserting new record or changing existing values in these tables. The update on marketing can be done on inventory by updating the genre, gender or age group information by the update query form.
* Promotional campaigns: When a new event is registered in the database it can be entered into the system by updating the Promotion Event table using the update query form. If the manager needs to view the information about the events he/she can use the query form specifying different conditions necessary.
* Sales statistics: When the manager needs to know the trend of sales in certain time range, age group, and genre and so on he/she can query using the specific conditions to create the report with desired information. Using the arrows can view the trends more clearly.
* Development schedules. Whenever a product enters a new stage and finishes its last stage, its related record in the table can be updated to reflect its current status. The schedule can be represented by the Completed Percentage column whose value is reported by the employee responsible for its current status. The manager can always query the status of the product to check if it is on or behind schedule.
* Manufacturer timelines: When a product is outsourced, related tables such as Outsourcing Invoice and Outsourced Product will be updated to record information about the date, manufacturer and the product. The date of the invoice can be viewed as the start date of the schedule set by the manufacturer and its planned due date is also recorded. The manufacturer reports its progress regularly and the attendants can update the information of related tables from those provided by the manufacture. The manager can always check the timeline using the query form to view a detailed report about the timelines and the products.

# Task6: SQL Queries



## Sales reports



### Report by age group

use project;

select tblgameplayed.AgeGroupID,tblagegroup.`GroupDescription`,sum(tblsalesitem.ItemQuantity) as Quantity,sum(tblsalesitem.ItemQuantity)\*`ProductPrice` as Turnover

from tblinventory,tblagegroup,tblgameplayed,tblsalesitem

where tblinventory.ProductISBN=tblgameplayed.ProductISBN and

tblinventory.productISBN=tblsalesitem.ProductISBN and

tblgameplayed.AgeGroupID=tblagegroup.AgeGroupID

group by tblagegroup.AgeGroupID

order by `Turnover` desc

;

Because this database assumes a product can have more than one age group, the quantity and turnover in the result contains some kind of duplication across different groups. But it still has business meaning for analyzing the trend of age structure of players. The screenshot shows the result:

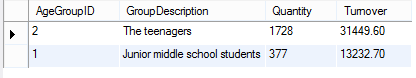


Figure – Report by age group

### Report by gender

Similar to the above, the SQL code is like following:

use project;

select tblgameplayed.genderID,tblgender.`genderName`,sum(tblsalesitem.ItemQuantity) as Quantity,sum(tblsalesitem.ItemQuantity)\*`ProductPrice` as Turnover

from tblinventory,tblgender,tblgameplayed,tblsalesitem

where tblinventory.ProductISBN=tblgameplayed.ProductISBN and

tblinventory.productISBN=tblsalesitem.ProductISBN and

tblgameplayed.genderID=tblgender.genderID

group by tblgender.genderID

order by `Turnover` desc

;

The screenshot shows the result:

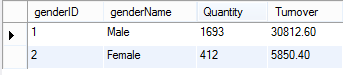


Figure – Report by gender

For the same reason, there is also duplication across the two genders here by aiming at both genders for one product.

## Inventory reports



### Report by specific genre

The SQL code is like following:

use project;

select tblgenre.GenreName,tblinventory.ProductISBN,`ProductName`,`ProductPrice`,`ProductQuantity`,`RetailPrice`,`LicenseFee`,`DistributionName`,`SupplierName`, `IPRType`,`ProductDescription`

from tblinventory,tblipr,tbliprtype,tblsupplier,tblgenre,tbldistribution

where tblinventory.IPRID=tblipr.IPRID

and tblipr.IPRTypeID=tbliprtype.IPRTypeID

and tblsupplier.SupplierID=tblipr.SupplierID

and tblgenre.GenreID=tblinventory.GenreID

and tbldistribution.DistributionID=tblinventory.DistributionID

and tblgenre.GenreName='Action'

order by `ProductName`

;

The screenshot shows the result:

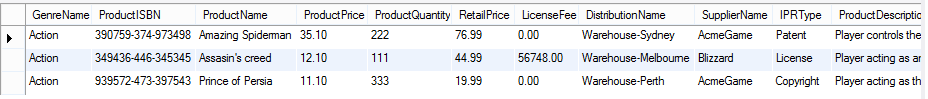


Figure – Report by specific genre

The code displays the inventory for products that belong to genre “Action”. The Description field is squeezed because there is not enough space in the workbench to show all of it.

### Report by specific price range

Similar to the above, the SQL code is like following:

use project;

select `ProductPrice`,GenreName,tblinventory.ProductISBN,`ProductName`,`ProductQuantity`,`RetailPrice`,`LicenseFee`,`DistributionName`,`SupplierName`,`IPRType`,`ProductDescription`

from tblinventory,tblipr,tbliprtype,tblsupplier,tblgenre,tbldistribution

where tblinventory.IPRID=tblipr.IPRID

and tblipr.IPRTypeID=tbliprtype.IPRTypeID

and tblsupplier.SupplierID=tblipr.SupplierID

and tblgenre.GenreID=tblinventory.GenreID

and tbldistribution.DistributionID=tblinventory.DistributionID

and `ProductPrice` > 10.00

and `ProductPrice` < 30.00

order by `ProductPrice`

;

The code displays the inventory of products that belong to the price range of (10.00, 30.00) (open interval). The screenshot shows the result:

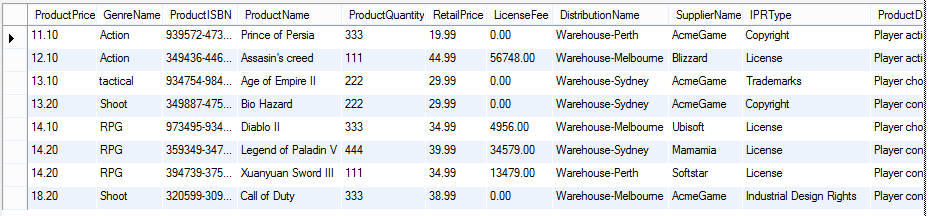


Figure – Report by specific price range

### Most profitable games (highest to lowest)

The SQL code is like following:

use project;

select (sum(`ItemQuantity`)\*`ProductPrice`) as Profit,

tblInventory.ProductName as Name

from tblSalesItem,tblInventory

where tblSalesItem.ProductISBN=tblInventory.ProductISBN

group by tblInventory.ProductISBN

# “group by” keywords indicates the results are listed in a way where the column following won’t have duplicate record which is why the result of function can’t be used here

order by Profit desc

;

The screenshot shows the results:

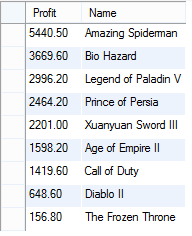


Figure – List of most profitable games

### Total sales by month and year

The SQL code is like following:

use project;

select sum(`ItemQuantity`\*`ProductPrice`) as Turnover,

tblsalesinvoice.MonthID,MonthName,tblsalesinvoice.YearID,YearNumber

from tblSalesInvoice,tblSalesItem,tblInventory,tblmonth,tblyear

where tblSalesInvoice.InvoiceID=tblSalesItem.InvoiceID and

tblSalesItem.ProductISBN=tblInventory.ProductISBN and

tblMonth.MonthID=tblSalesInvoice.MonthID and

tblYear.YearID=tblSalesInvoice.YearID and

MonthName =’September’ and

YearNumber = 2000

;

The screenshot shows the result:



Figure – Sales by month and year

### Most successful retailer

The SQL code is like following:

use project;

select max(`RetailerSale`) as RetailerSale, a.CustomerID as RetailerID,a.CustomerName as Retailer

#because “select” statement selects items from the “from” statement, the only way to reference a column in the database is listing it in the sub-query and reference it using the alias of the sub-query

from (

select sum(`ItemQuantity`\*(`RetailPrice`-`ProductPrice`)) as RetailerSale,tblsalesinvoice.CustomerID,CustomerName

#for every product the netprofit timed by the quantity sold is total earned profit which will be summed together for every customerID indicated by "group by" keyword

from tblSalesItem,tblInventory,tblSalesInvoice,tblcustomer

where tblSalesItem.ProductISBN=tblInventory.ProductISBN and

tblSalesItem.InvoiceID=tblSalesInvoice.InvoiceID and

tblsalesinvoice.CustomerID=tblcustomer.CustomerID

group by tblSalesInvoice.CustomerID

order by RetailerSale desc

) as a

# “a” is an arbitrary name for the sub-query because alias is necessary for sub-query

#the sub-query itself returns a table listing retailers’ sales from highest to lowest

;

The sub-query returns a temporary view:

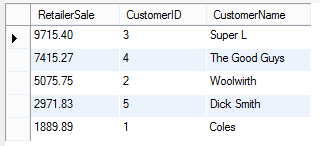


Figure – Sub-query for most successful customers

The result of the query is like following:



Figure – Most successful retailer

# Recommendation

In order to strengthen the cooperation of different departments in Acme Game Company the author recommends that the new database should be implemented as soon as possible to adapt to the situation of fast-growing demand in the game market. The new database can optimize the inventory control, update sales and marketing information efficiently, analyze data from promotional campaigns, display sales statistics easily and track schedules of development and manufacture of new products. The new system can replace the old one in a short time without the need of phase-by-phase installation or parallel installation because it is only a relatively small system. The system can be scaled up to contain more and more data depending on the need of the company.

# Conclusions

This report analyzed the requirement of the Acme Game Company for designing the new database system and then listed the main entities the new database should have. It gave the detailed process of development for the database with E-R diagrams from first normal form to third normal form and explained the diagrams in detail. It also described how the user interface should be like and how the database should work. Finally it gave the codes for creating the database, inserting sample data and executing sample queries. The author at last recommends the new system should be implemented immediately.

# Appendix



## Screenshots for sample data inserted

The following are screenshots demonstrating the sample data is inserted:

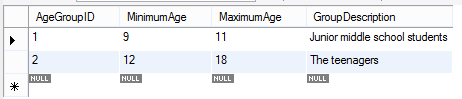


Figure – tblAgeGroup

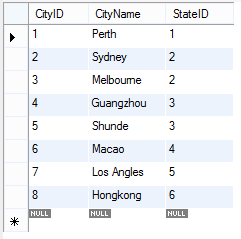


Figure – tblCity

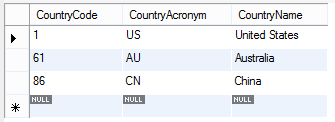


Figure – tblCountry

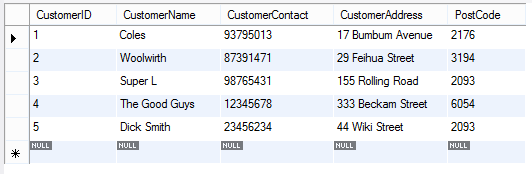


Figure – tblCustomer

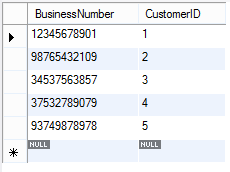


Figure – tblCustomerBusiness

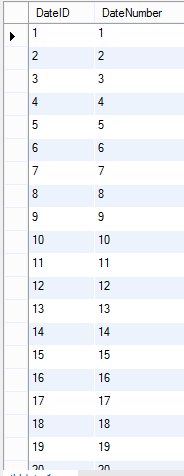


Figure – tblDate

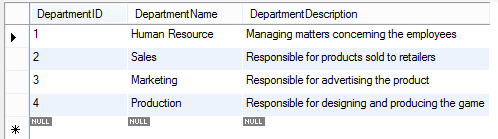


Figure – tblDepartment

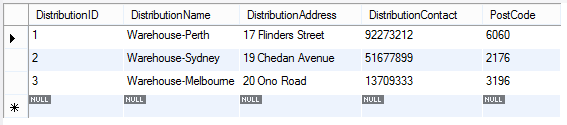


Figure – tblDistribution

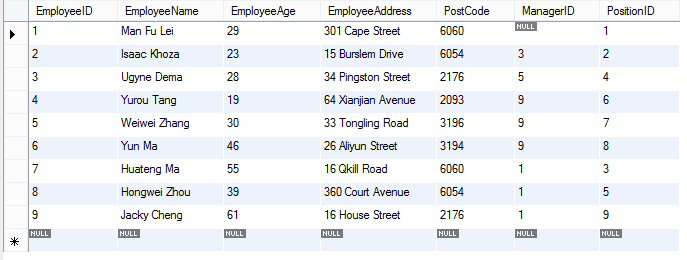


Figure – tblEmployee

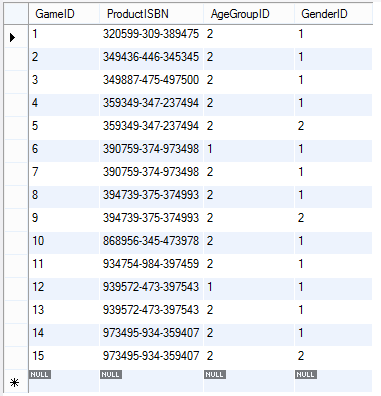


Figure – tblGamePlayed

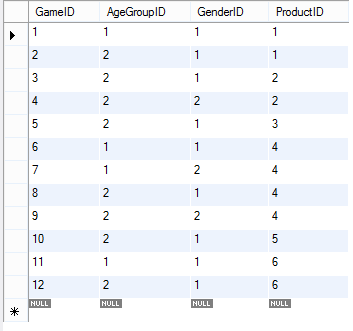


Figure – tblGameTested

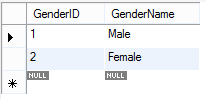


Figure – tblGender

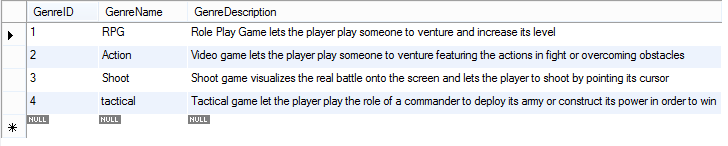


Figure – tblGenre



Figure – tblInventory

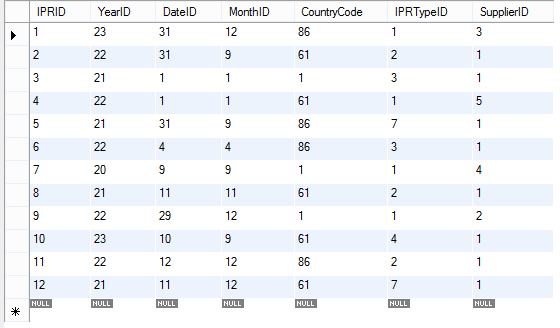


Figure – tblIPR

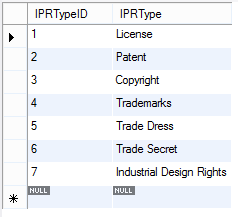


Figure – tblIPRType

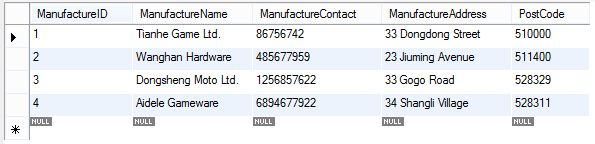


Figure – tblManufacture

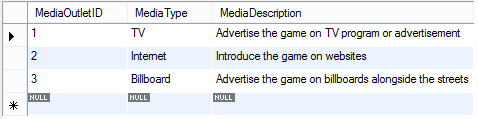


Figure – tblMediaOutlet



Figure – tblMonth

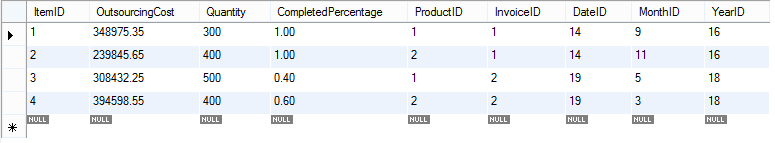


Figure – tblOutsourcedProduct

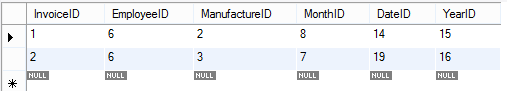


Figure – tblOutsourcingInvoice

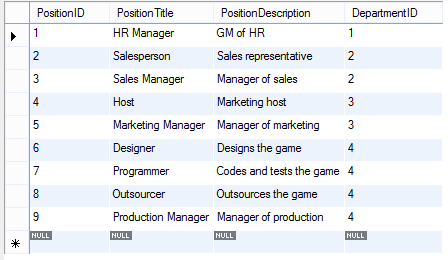


Figure – tblPosition

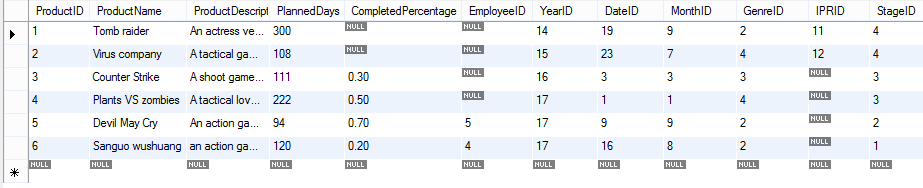


Figure – tblProduction

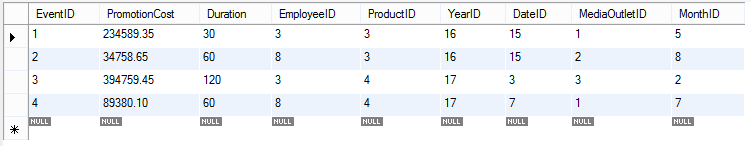


Figure – tblPromotionEvent

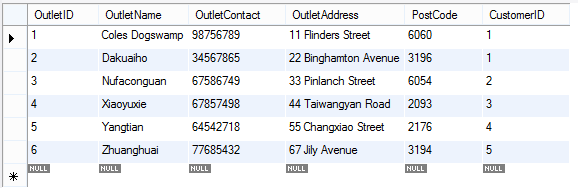


Figure – tblRetailOutlet

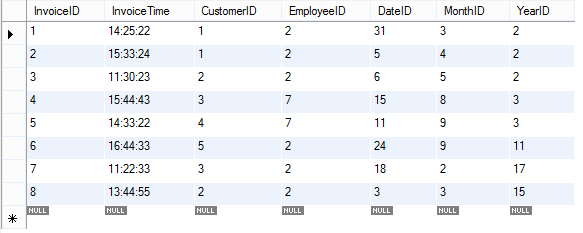


Figure – tblSalesInvoice

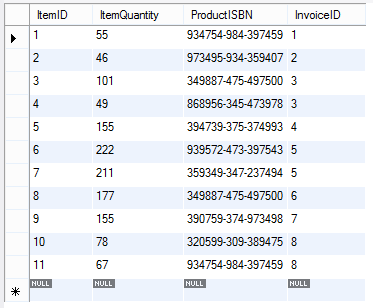


Figure – tblSalesItem

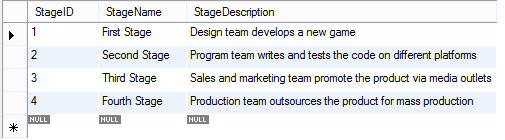


Figure – tblStage

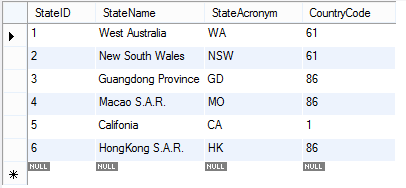


Figure – tblState

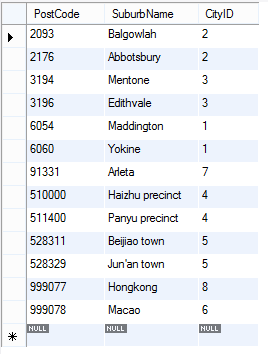


Figure – tblSuburb

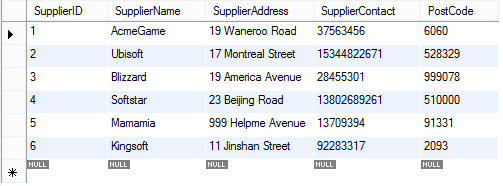


Figure – tblSupplier



Figure – tblYear

# List of References

11.2.1 Integer Types (Exact Value) - INTEGER, INT, SMALLINT, TINYINT, MEDIUMINT, BIGINT. (n. d.). In *MySQL.com*. Retrieved Sptember 27, 2014 from <http://dev.mysql.com/doc/refman/5.1/en/integer-types.html>

Ahsan, M. (2010). *Using MySQL Stored Procedure to create sample data*. Retrieved September 24, 2014 from <http://thinkdiff.net/mysql/using-mysql-stored-procedure-to-create-sample-data/>

Anshul. (2011 March 25) RE: MySQL: Reorder/Reset auto increment primary key? [Online Forum Comment] Retrieved September 25, 2014 from <http://stackoverflow.com/questions/740358/mysql-reorder-reset-auto-increment-primary-key>

Hoffer, J., Ramesh, V. & Topi, H. (2011). *Modern database management*. Upper Saddle River, N.J: Prentice Hall.

Intellectual property. (n. d.). In *Wikipedia*. Retrieved September 24, 2014 from <http://en.wikipedia.org/wiki/Intellectual_property>

jp. (2009 August 22) RE: How to fix: MySQL Duplicate entry for key PRIMARY on Auto\_Increment. [Online Forum Comment] Retrieved September 24, 2014 from <http://www.softwareprojects.com/resources/programming/t-how-to-fix-mysql-duplicate-entry-for-key-primary-o-1844.html>

List of country calling codes.(n. d.). In *Wikipedia*. Retrieved September 24, 2014 from <http://en.wikipedia.org/wiki/List_of_country_calling_codes>

SQL Data Types for Various DBs. (n. d.). In *w3school.com*. Retrieved September 27, 2014 from <http://www.w3schools.com/sql/sql_datatypes.asp>

# Glossary

**Database:** set of software programs providing all users with access to all the data

**Relational database model:** stores information about both the data and how it is related

**Logical database design:** a type of data model showing a detailed representation of some or all of an organization's data, independent of any particular data management technology, and described in business language

**Structured Query Language (SQL):** a special-purpose programming language designed for managing data held in a relational database management system (RDBMS)

**Query:** A computer language used to make queries into databases and information systems

**User Interface Design:** The design of websites, computers, appliances, machines, mobile communication devices, and software applications with the focus on the user's experience and interaction

**Intellectual Property Rights (IPR):** Legally recognized exclusive rights to creations of the mind

**Entity Relationship (E-R) diagrams:** Data model for describing the data or information aspects of a business domain or its process requirements, in an abstract way that lends itself to ultimately being implemented in a database such as a relational database

**First normal form:** A relation is in first normal form if thedomain of each attribute contains only atomic values, and the value of each attribute contains only a single value from that domain

**Third normal form:** All the attributes in a table are dependent on the primary key and only the primary key

**Attribute:** Also called field which is the metadata describing the object from a specific aspect

**MySQL workbench:** open-sourced software which uses MySQL as the language norm

**SQL: 200n:** A future standard of SQL released in 2003

**American Psychological Association (APA):** Academic format specified in The Publication Manual of the American Psychological Association, a style guide that offers academic authors guidance on various subjects for the submission of papers to the publications of APA

**Entity:** Objects that need to be represented in E-R diagram

**International Standard Book Number (ISBN):** Unique numerical number identifying the digital object itself in this world

**Associative entity:** A kind of entity in E-R diagram which depends on other entities to exist

**Many-to-many relationship:** A relationship in E-R diagram saying an object can have multiple instances of another object for interaction while the other one has the same relationship to it too

**Composite key:** More than one key is used as the primary key of the entity

**Primary key:** The attribute uniquely identifying every instance of the entity

**Second normal form:** Every non-key attribute is fully dependent to the full set of primary keys

**Foreign key:** A field (or collection of fields) in one table that uniquely identifies a row of another table

**Ternary relationship:** A relationship in E-R diagram that involves at least three entities to form the relationship

**Normalization:** The process of reducing redundancies in relational database model

**Functional dependency:** From every value of one attribute the value of another attribute can be determined non-mathematically

**Redundancy:** The degree to which the records in the tables are duplicated in several fields

**One-to-one relationship:** A relationship in E-R diagram where one entity can only interact with one instance of another entity at a time while the other entity holds the same relationship to it

**Non-key attribute:** Attributes other than the primary key

**Unary relationship:** A relationship in relational database model where the attribute of the entity references the primary key of itself

**Thin client mode:** A mode in database design where the client is free of additional software to operate towards the database

**Three-tier architecture:** A client–server architecture in which presentation, application processing, and data management functions are physically separated

**Web server:** The component which provides the presentation function for the client in database architecture

**Application server:** The component which provides database server connection, access and data-related presentation for clients in database architecture

**Java Servlet Pages (JSP):** A technology that helps software developers create dynamically generated web pages based on HTML, XML, or other document types

**Driver:** The piece of software which provides the connection from the software to the hardware

**Procedure:** Another name for function that is stored in the database server to provide pre-defined function when called explicitly

**Fat server:** Indicating most software and functions are stored in the server side to reduce the workload of the client and the network traffic

**Phase-by-phase installation:** A method of system installation where the new system is implemented phase-by-phase to provide enough preparation for the implementation of new system

**Parallel installation:** A method of system installation where the new system coexists with the old system until the old one can be completely replaced

# Index

A

Acknowledgements, 2

Acme Game, 1

Acme Game Company, 1

Action, 48

Age group, 6

ID, 6

Ahsan, 46

American Psychological Association (APA) format, 1

APA. *See* American Psychological Association (APA) format

Appendix, 72

Associative entity, 7

Authorization, 1

B

Billboard, 49

Business Number, 8

C

Califonia, 50

City, 6

Completed percentage, 10

Composite key, 7

Conceptual Database Model, 12

Conclusion, 71

Connection, 60

Country, 6

Customer, 8

Address, 8

Contact, 8

Name, 8

D

Database, IX, 25

Delimiter, 46

Department, 9

Design, 49

Development schedules, 4, 61

Distribution, 6

ID, 6

Name, 6

Driver, 60

Due Date, 11

Duration, 11

E

Employee, 8

Address, 9

Age, 9

Name, 8

Entity, V, 1, 5, 13, 17, 71, 83, 84

Entity Relationship (E-R) diagrams, 1

first normal form, 1

second normal form, 7

third normal form, 1

E-R. *See* Entity Relationship (E-R) diagrams

Event, 11

Date, 11

ID, 11

Media, 11

Executive summary, IX

F

Foreign key check, 46

Form, 57

sample, 57

G

Game, 7

ID, 7

GameTested, 18

Gender, 6

ID, 6

Name, 6

Group Description, 6

Guangdong Province, 49

Guangzhou, 50

H

Hoffer, 60

Hongkong, 50

HongKong

S.A.R., 50

Human Resource, 49

I

Industrial design right, 7

Integer, 19

Intellectual Property Rights, 1

Internet, 49

Introduction, 1

Inventory, 7

Inventory control, 4, 61, 70

Invoice, 9

Date, 9

ID, 9

Month, 9

Time, 9

Year, 9

IPR, 7, *See* Intellectual Property Rights

Expiry Date, 7

ID, 7

Type, 7

Item, 9

ID, 9

Quantity, 9

J

Java Servlet Pages, 60

jp, 32

JSP. *See* Java Servlet Pages

L

List of references, 82

Literature review, 4, 13

Logical Database Model, 15

Logical design, IX

Los Angles, 50

M

Macao, 50

S.A.R., 49

Manufacture, 9

Address, 10

Contact, 10

ID, 10

Name, 10

Manufacturer timelines, 61

Manufacturer's timelines, 4

Marketing, 49

Maximum age, 6

Melbourne, 50

Minimum age, 6

MySQL workbench, 1

N

Navigation system, 59

New South Wales, 49

Null values, 5

O

Online Research, 3

Open interval, 66

Outlet, 8

Address, 8

ID, 8

retail, 8

Outsourced product, 11

Outsourcing Cost, 11

Outsourcing invoice, 10

P

Patent, 7

Perth, 50

Planned Days, 11

Position, 9

Title, 9

Postcode, 6

Primary key, 7

Procedure

stored, 46

Product, 7, 10

Description, 7, 10

Genre, 10

ID, 7, 10

ISBN, 7

Name, 7, 10

Price, 7

Quantity, 7

Stage, 10

Production, 10, 49

Program, 49

Promotion Cost, 11

Promotion event, 11

Promotional campaigns, 4, 61, 70

Purpose, 1

Q

Quantity, 11

Query, IX

form, 61

update, 60

R

Ramesh, 60

Recommendation, 70

Redundancy, 4, 7, 11, 13, 17

Relational database model, IX, 5, 84

Relationship

many-to-many, 7

Report, 58

sample, 59

Requirement, 3, 4

Retail Price, 7

RPG, 48

S

Sales, 49

Sales and marketing updates, 4, 61

Sales Invoice, 9

Sales item, 9

Sales statistics, 4, 61, 70

Sample records, 46

Scope, 1

Server

application, 60

database, 60

fat, 60

web, 60

Shoot, 48

Shunde, 50

SQL. *See* Structured Query Language

200n, 1

Stages, 4, 10, 13

Start Date, 10

State, 6

Structured Query Language, IX

Suburb, 6

Supplier, 5

Address, 5

Contact, 6

ID, 5

Name, 5

Sydney, 50

System Design, 62

T

Tactical, 48

Terms of Reference, 1

Thin client mode, 60

Thornlie, i

Three-tier architecture, 60

Tier

first, 60

second, 60

third, 60

Topi, 60

Trademark, 7

Turnover, 7

TV, 49

U

User interface design, IX, 57

V

Varchar, 19

W

West Australia, 49