Information Systems Assignment Part4/Final Submission

Joe Davies (17029907)

James Brito Alvarado (18012469)

Oliver Bevington (18040156)

Nathan Ainsley (18028669)

Date Submitted: 15/03/19

Course Tutor: Stephen Gordon

Year: one

Contents

[Summary/Introduction 6](#_Toc3555396)

[Part 1 6](#_Toc3555397)

[Introduction/Summary 6](#_Toc3555398)

[Information on part 1 changes/reparations 7](#_Toc3555399)

[Problems with the Paper-based system. 8](#_Toc3555400)

[System Requirements. 9](#_Toc3555401)

[Types of data enquiries and management reports. 10](#_Toc3555402)

[Outline several data enquiries and management reports that you think will be of use to Ray’s Rentals. 12](#_Toc3555403)

[Analysis Reports 12](#_Toc3555404)

[Key Target Reports 13](#_Toc3555405)

[Exception Reports 14](#_Toc3555406)

[Ad-hoc Reports 15](#_Toc3555407)

[Part 2 16](#_Toc3555408)

[Introduction 16](#_Toc3555409)

[Use Case Diagram (MOSCOW Prioritisation) 17](#_Toc3555410)

[Use Case Diagram Commentary and though processes 18](#_Toc3555411)

[Use Case Specifications (Core Use Cases) 19](#_Toc3555412)

[Selling a Bike by James Brito Alvarado (18012469) 19](#_Toc3555413)

[Dealing with Enquiries by Joe Davies (17029907) 20](#_Toc3555414)

[Buying a New Bike by Oliver Bevington (18040156) 21](#_Toc3555415)

[Renting a Bike by Nathan Ainsley (18028669) 22](#_Toc3555416)

[Top Down ERD 24](#_Toc3555417)

[RDA with Bottom up ERD 25](#_Toc3555418)

[Finalised group ERD 28](#_Toc3555422)

[Final ERD Summary 29](#_Toc3555423)

[Conclusion 29](#_Toc3555424)

[Part 3 30](#_Toc3555425)

[Introduction 30](#_Toc3555426)

[What we have learnt 30](#_Toc3555427)

[Nathan Ainsley: 30](#_Toc3555428)

[Joe Davies: 30](#_Toc3555429)

[James Brito Alvarado: 31](#_Toc3555430)

[Ammended ERD 32](#_Toc3555431)

[Data Dictionary 33](#_Toc3555432)

[SQL Developer Query Screenshots 38](#_Toc3555433)

[Query/Report 1. 38](#_Toc3555434)

[Query/Report 2. 38](#_Toc3555435)

[Query/Report 3. 39](#_Toc3555436)

[Query/Report 4. 40](#_Toc3555437)

[SQL Developer script output. 40](#_Toc3555438)

[What people have learnt total 44](#_Toc3555440)

[Appendix 45](#_Toc3555441)

[References 45](#_Toc3555442)

[Piazza Posts 45](#_Toc3555443)

[Part 1: 45](#_Toc3555444)

[Part 2: 46](#_Toc3555446)

[Part 3: 48](#_Toc3555451)

[Part 4: 49](#_Toc3555452)

[Presentation 50](#_Toc3555453)

[Copy of script file 64](#_Toc3555455)

# Summary/Introduction

When presented with the task of creating a replacement database system for the old paper based one that rays rentals had we first had to identify key problems with the current system.

* Paper based made it, so it was bulky
* No way of properly organising the system
* Records where often lost
* Orders where misplaced

So, in this report we set out to both create an entirely new system for ray to use but also to fix those core problems by doing so. This report covers all the aspects of the process we went through to create this system which included:

* Identifying problems with the current system.
* Identifying requirements for the new system.
* Identifying types of queries that will be integrated into the new system.
* Creating a Use Case Diagram to Identify how each actor will interact with the system.
* Creating Use Case Specifications which allowed us to identify key processes that the system had to undertake.
* Creating a Top Down Entity Relationship Diagram.
* Creating a Relational Data Analysis and a bottom down Entity Relationship Diagram.
* Creating a finalised Entity Relationship Diagram.
* Creating a Data Dictionary to allow us to implement the system.
* The Oracle SQL system in full.
* Outputs of the Sql queries located in the system and how they work.

In All this Report will show you how we went about creating the new system for Rays Rentals and our mindset whilst doing so.

# Part 1

## Introduction/Summary

In the following part of our assignment, the main focus will be on the research and preparation going into creating a new system for Ray’s Rental Bike Shop. With the current system being paper based, there are a mass of problems that are very easy to pick up when reading the case study. It is quite obvious that some sort of online implementation using a database is necessary to upkeep the business, especially if it is to grow some more. This part of the assignment will discuss the main problems/issues with his existing system, from which one can contemplate and use these problems to help discuss and build a new system.

Before going straight into building a new system from scratch without any plan, there needs to be some background work, preparation, and research. Without all of this, it would 100% not be implemented correctly, and would take a lot more time to do. Like writing a dissertation with no plan. The following work in part one will set out new system requirements, mainly on the functional side, as we need to know what the system needs to be able to do. Following this there will be a research section of the different types of management reports that we think will be of use to Ray’s rentals. These are not implemented in ORACLE yet, and are only ideas for our future projects.

We concluded that all four main types of management report will be of use/could be of use to Ray’s new system. All do different things, but all offer a concise report that conveys a lot about the current workings of the business. The last part of this section will discuss ideas for actual reports we might implement later on, hopefully with some improvements as our knowledge of Information Systems and the SQL Language develops.

## Information on part 1 changes/reparations

As informed by our tutor, we have been allowed to add changes to our part one to improve the mark. In the following part one section, any text that has been updated/altered for a (hopefully) increase in mark has been highlighted in yellow. For other changes that can’t be highlighted (pictures/figures etc.) are detailed below:

1. All Figures in part one have now been correctly labelled as it was brought up that they were not.
2. Figure 2.2 in the Data Enquiries Section has been altered to show a better version of what a Key Target report would look like, due to the fact that are last one showed a projected/predicted profit next month, which was overly complicated and could not really be easily implemented by first years like ourselves.

## Problems with the Paper-based system.

Although the current system works and provides good customer service the process in how this is completed is very complex and there are many areas in which a fault could take place. Firstly, each bike is to have a Bike Record while this is good to allow Ray and the team to keep track of every bike and all of the details, it brings the issue of having approximately 150 pieces of paper for the bikes alone. This raises the issue of a lengthy process to even find one bike’s record. Another problem in the current system is the amount of information taken down on hand-written notes and passed around, none of this is recorded and can easily be lost or mismatched. If staff members are taking down customer information over the phone and they then forget about the note they wrote that on, the bike is never reserved for the customer, which is a major issue where the only safe guard is the employee’s memory. Another key document is the Rental Record which is several pieces of paper pinned together. The main issue with this is the documents being lost with customer’s information on them, the documents hold personal information which anyone with any authority could get a hold of.

The current process and thinking of the system and data taken down is not bad but the way the system functions and how it is paper based is very poor. Finally, due to the current system relying on just paper this would be looked to be changed to computers controlling all the media and data passes through or an alternative of documents and a better storage system. These are both better alternatives reducing the amount of different papers being passed through Rays Rentals.

A problem regarding the service of bikes is the fact that Ray draws up a hand-written list of the bikes that have not been serviced for a month, from which the two technicians work through in the following week. This is a major issue in the fact that they have no real system to prioritise the bikes that need servicing, because essentially with their current system every bike is being serviced an equal amount of times when really, to work efficiently it should not work like this. Some bikes will be rented more than others, and these high demand bikes could be at the bottom of the queue, so it is entirely possible that there would be bikes that are in need (because of high demand) that are unavailable to rent due to less in demand bikes being ahead of them in the queue.

A problem regarding the repair of the bikes is that there is very often parts that are over-ordered, and some that are under-ordered, meaning some parts will go rusty and also that some bikes are left lying around for a long time due to the fact that the parts staff have not even ordered the relevant parts. This has an easy solution, by implementing a database, it would be a lot easier to keep track of how many parts are in stock, to avoid over ordering and to allow the quick repair of bikes that have been left to sit for too long.

All of these issues will affect the business in a multitude of ways. The problem with repairs and services will affect the rate of which bikes are rented out, which will in turn affect the amount of profit. All of this could be solved with a new system implemented with an ORACLE Database.

## System Requirements.

Ray will need the new system to be able to notify him when a bike service is due. In addition, when it is near its 2-year warranty date so he can sell them. It is important that Ray has access or is notified whether there is any faults with the bikes so they can be repaired. Currently Ray has all his rental forms, general maintenance history and mailing lists completed by hand. If this were done by Access it would speed up the process this is because Ray and his team will be able to input new data into the database and it will auto complete more of the admin such as the; date and time, pricing, renters address and type of bike that is being used. Ray could also use the data to change his buying strategy and see what bikes are more popular than others.

With the current system being fully paper based, lots of problems arise regarding basic functionality and arguably more importantly, the speed of which the system runs. Moving the whole system online is exactly what Ray needs, and there are certain requirements that should be met to ensure that when the new system is complete and ready to be rolled out, it runs efficiently and without error. The system will run on an ORACLE Database using SQL Developer to allow for the linking and intertwining of tables for maximum reusability, and to make retrieving information regarding the system accurate and speedy.

The Four main aspects of the system will be, the Customer, The Employees, The Bike Manufacturer, and the Bike. All of these “actors” need to be able to perform certain tasks. Below is a bullet point list describing what each actor needs to be able to do to provide Ray with a basic functional system:

Customer needs to be able to:

∙ Enquire about bikes

∙ Ring to report faults

∙ Hire a bike

Employees need to be able to:

∙ Deal with enquiries

∙ General Admin

∙ Order new parts

∙ Perform maintenance on bikes

∙ Update relevant records when necessary (maintenance, rentals, parts etc)

The Bike manufacturer needs to be able to:

∙ Buy a bike

∙ Sell a bike

∙ Update contact details and other relevant information

The Bike needs to be able to:

∙ Have its relevant details updated in its related tables

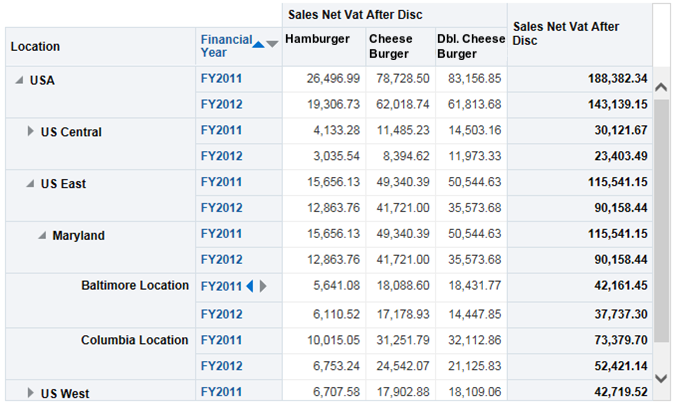
## 

## Types of data enquiries and management reports.

A data enquiry is a question/query relating to the organisation or business made by someone, usually a potential customer or a member of staff. Data enquiries relating to Ray’s Rentals that customers may query might include; Price, price per time ratio, type of bike, brand of bike, accidental damage information, etc.

A management report is a concise report that management leaders use to evaluate and read into their business operations and how successful they are.

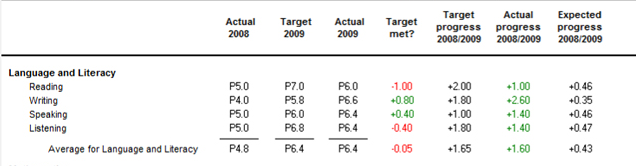
Analysis Reports are used to give a summary of data for a particular period, whether that is for just that period or to compare a period of time with another. The summary of data could be about anything, a common use of this type of report for example is to look at how profitable certain aspects of the system are. It is a more in depth report as opposed to others and can show a lot of information about a system and its performance. An example of an analysis report geared towards net profit is shown in figure 1.0 (Analysis Reports, Oracle Hospitality Reporting and Analytics User Guide)



*Figure 1.0*

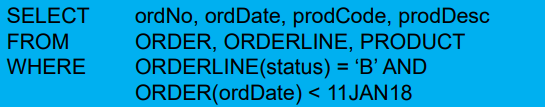
Exception reports are used to show anomalies outside of the normal reports a company gets. These types of reports are used to eliminate exceptions like itself and overall contribute to making the business more reliable and consistent by minimalizing future “exceptions”. An example is shown below (Exception Report Template, date: unknown)



Key target Reports are very useful reports that compare expected or predicted sales/figures against the actual number given at that specific time. These are used to show which periods are underperforming and they can work from that to figure out why. An example is shown in figure 1.1. (Analysis of actual progress versus target progress, date: unknown)

*Figure 1.1*

Ad-hoc Reports are the most basic form of management report since they are just one off queries not designed for in depth analysis. They are to get quick and easy answers for general questions they may have for a specific point of business. An example is shown in figure 1.2. (Information systems Lecture slides, MMU, 2018)



*Figure 1.2*

## 

## Outline several data enquiries and management reports that you think will be of use to Ray’s Rentals.

### Analysis Reports

**Report on how many rentals there were in the past 2 months and compare it to profit in those given months.**

* Data Needed:

-Number of rentals in this month, and last month

-Average number of rentals per month

-Amount made this month and last month

-Profit made this month and last month

-Average amount made, of profit, and spent by month

* Time frame

- 2 months, and then will have a monthly average from past years’ worth of reports, which will automatically build up as reports are generated month by month.

* Visual: Figure 2.0



*Figure 2.0*

* How it might be displayed:

- output a text document.

**Generate a report on all rentals from a date**

* Data needed:

-Rental ID

-Date and length of rental

-Date returned

-Price of rental

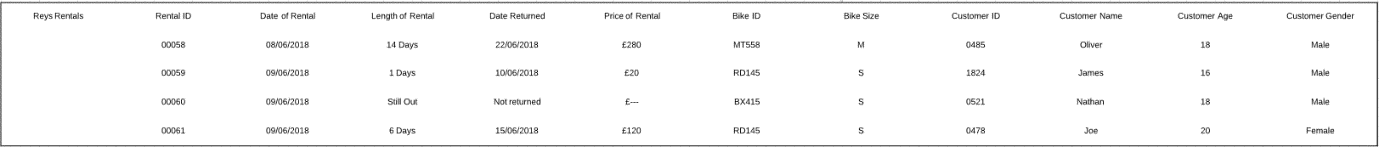
-Bike ID, size and classification

-Customer ID, name, age, and gender

* Time frame:

- The inputted date up to the system time

* Visual: Figure 2.1



*Figure 2.1*

* How it might be accessed:

-Report would be accessed by entering a date that you want to be the start. Then outputted to text document

### Key Target Reports

**Generate a report on profit made**

* Data needed:

-Profit made, this month and last

-Average profit made

-Expected profit next month

-Rentals this month and last

-Average, expected rentals

* Time frame:

-Past three months, including current one

* Visual: Figure 2.2

Goal

*Figure 2.2*

* How might it be accessed?

-Outputting a text document

### Exception Reports

**Generate a report on bikes booked that haven’t had a service in (x) months**

* Data needed:

-Bike ID

-Service ID

-Last serviced

-Service type

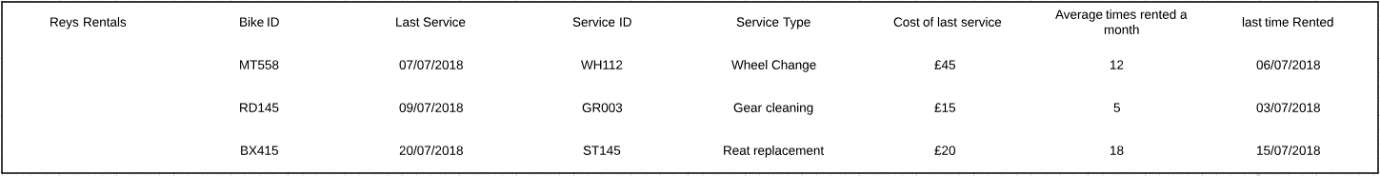
-Cost of last service

-Average times rented a month

-Last time rented

* Time frame:

-The report should cover the period from the given date up until the system time

* Visual: Figure 2.3

*Figure 2.3*

* How is it generated?

- outputting a text.

**Report on a single bike being used (x) amount of times in a month**

* Data needed:

-Bike; ID, Category

-Amount of times rented

-Profit from rentals

-Amount of times sorted

* Time frame:

-The report will cover a period of one month

* Visual: Figure 2.4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bike ID | Amount Rented | Different Renters | Profit | Repairs (Y/N) |
| BK-123 | 24 | 20 | £730 | Y |
| BK-28 | 21 | 12 | £620 | N |
| BK-88 | 30 | 28 | £760 | Y |
| …. | …. | …. | …. | …. |

*Figure 2.4*

* How is it generated?

-The user can type in the amount they want the report to search for and will output any bike that has had more than that number of rentals per month.

### Ad-hoc Reports

**Search for a bike via Model/size/classification**

* Data needed:

-Bike ID, model, size, classification

-Rented or in stock

-Last time rented

-Last time serviced

-Price

* Time frame:

-As this is a generic search for a bike the search won’t have a time frame to cover as it is looking for either specific records or groups of records

* Visual:

-The records found will be outputted into a textbox in the GUI

**Search for parts to view what’s in stock via Stock ID**

* Data needed:

-Part ID, name, type, price, stock level

* Time frame:

-The search would not cover a time scale as it will just show what is currently in stock via the stock ID which is searched for

* Visual:

-The search will be outputted to a text box in the GUI

* How might staff generate the report?

-The staff will type in the stock ID, name or size and it will return with the records it finds

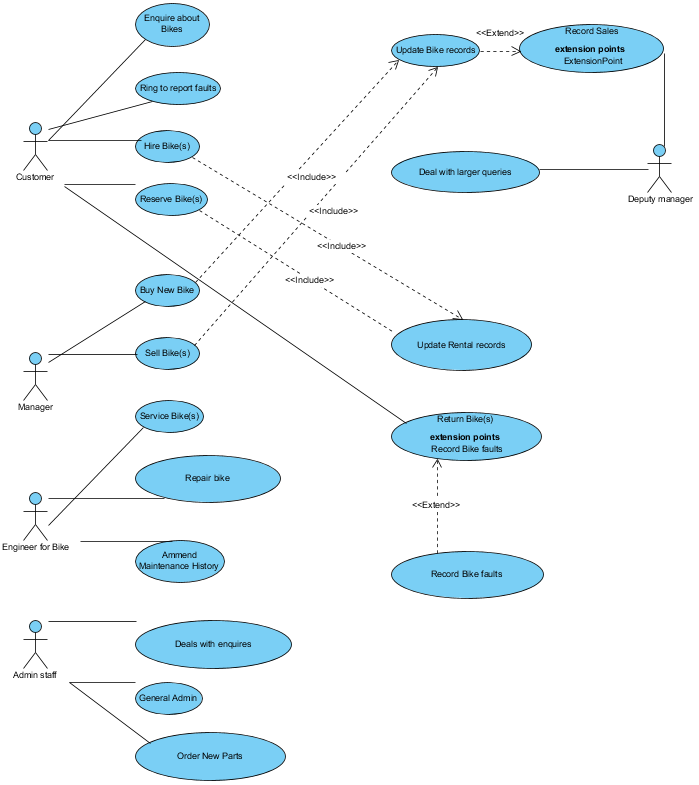
# Part 2

## Introduction

This report summarises all of the vital diagrams needed to implement our new system to replace Ray’s old one. First, each member of the team made the use case specifications. From this, we developed the use case diagram, identifying the actors and what they will achieve from the system. Once both the UCD and the use case specifications were complete, we then performed the RDA’s, normalising the system to the third form.

From the RDA’s, a bottom-up ERD was developed, along with a top-down ERD. From the two ERD’s, the finalised merged ERD containing ideas from both, was mocked up. Once we all looked over and tweaked the final ERD, and got feedback from staff

## Use Case Diagram (MOSCOW Prioritisation)



## Use Case Diagram Commentary and though processes

There were several crucial steps that went into developing the main UCD for the replacement system. First the actors that will be used are identified; the case study summary mentions five crucial/integral actors; Customer, Manager, Deputy Manager, Technicians, Admin staff. Once the actors are identified, the use of the actors (the use cases) are then identified. This was done by essentially making an organised table consisting of the actors down one column, their use cases in another, and the relevant dependencies (include, extend) down the last one. The mocked up table is shown in figure 1.0

|  |  |  |
| --- | --- | --- |
| **ACTOR** | **USE CASES** | **DEPENDENCIES** |
| Customer | 1. Ring and enquire 2. Reserve a bike🡪🡪🡪 3. Hire a bike🡪🡪🡪🡪🡪 4. Report bike fault 5. Return bike🡨🡨🡨🡨 | Update Rent Records(Include)  Update Rent Records(Include)  Record bike faults (Extend) |
| Manager | 1. Buy a new bike🡪🡪🡪 2. Sell a bike🡪🡪🡪🡪🡪 | Update Bike Records(Include)  Update Bike Records(Include) |
| Deputy Manager | 1. Deal with large enquiries 2. Record Sales | EXTEND |
| Bike Technicians | 1. Service bike 2. Repair bike 3. Amend maintenance history |  |
| Admin Staff | 1. Deal with enquiries 2. General Admin 3. Order new parts |  |

*Figure1.0*

Once the dependencies are identified you can link the ones that share a use case and from there, draft a UCD and connect them.

## Use Case Specifications (Core Use Cases)

## 

### Selling a Bike by James Brito Alvarado (18012469)

|  |
| --- |
| **Use Case: Selling a bike** |
| Owner: Manager |
| **Pre-Conditions** |
| After 2-3 years bike is serviced and then sold to a buyer who usually lives in the local village |
| **Post-Conditions** |
| Buyer is given all the service records and rental certificates and payment is received |
| **Primary Path** |
| 1. Owner provides bikes that have been owned for more than 2/3 years 2. A service is completed and the service record is updated 3. An enquiry is made and owner then provides details to a buyer 4. Bike is sold and delivered to buyer 5. Bike then is removed from database |
| **Alternate Path** |
| 1. Payment not received 2. Bike will not be sold to buyer and buyer is contacted 3. New buyer is found |
| **Notes** |
| https://i.gyazo.com/b5122018060c331d6343d51a4d621925.png |

### Dealing with Enquiries by Joe Davies (17029907)

|  |
| --- |
| **Use Case: Dealing with Enquiries** |
| Owner: Admin Staff |
| **Pre-Conditions** |
| . Customer has a query with a specific bike in mind  . Admin staff able to receive call  . Customer and admin staff have an email |
| **Post-Conditions** |
| . Customer is given an email with information on said bike(s) |
| **Primary Path** |
| . Customer contacts admin team about a bike enquiry  . Admin team deliver relevant information about the bike query  . Customer then either decides to rent the bike on the spot or go away and think about it |
| **Alternate Path** |
| . Customer contacts admin team about a bike enquiry  . Admin team inform them the bike is currently unavailable due to it already being in use or currently having a service  . Customer then either decides to reserve the bike for a future time on the spot or go away and think about it some more |
| **Notes** |
| https://i.gyazo.com/b5122018060c331d6343d51a4d621925.png |

### Buying a New Bike by Oliver Bevington (18040156)

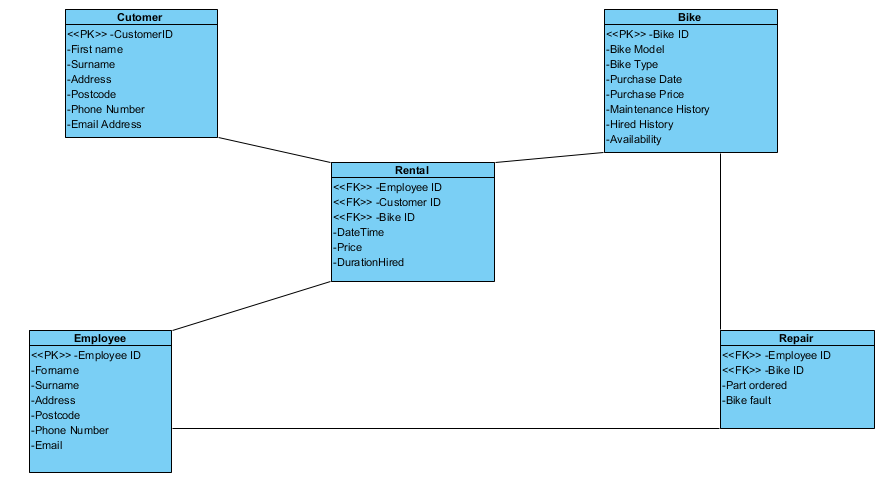
|  |
| --- |
| **Use Case: Buying a New Bike** |
| Owner: Ray |
| **Pre-Conditions** |
| 1. Order to be sent to the supplier 2. Bike to be available 3. Money for the bike to be sent to the supplier |
| **Post-Conditions** |
| 1. Bike identified and ordered 2. Payment to be received 3. Bike to be delivered to Ray |
| **Primary Path** |
| 1. Ray goes online to purchase a bike 2. Ray chooses a bike to buy 3. It is then checked and should show online if it is available to buy 4. The order form is then filled out giving the size of bike, delivery address and payment information 5. Payment to be sent 6. The order is then placed and received by the company 7. The supplier will receive the order giving the details of the ordered bike 8. The supplier will collect, or build said bike 9. This bike will then be packaged up 10. The bike is then sent over to Ray’s Rentals 11. Ray will then receive the bike at Ray’s Rentals 12. The bike is then unpackaged by Ray 13. The bike is checked out, given an ID number and added to the database. |
| **Alternate Path** |
| 1a. Ray visits a bike shop  3a. The Bike is not available to buy  5a. Payment to be denied  11a. Bike not to be received |
| **Notes** |
| **https://i.gyazo.com/b5122018060c331d6343d51a4d621925.png** |

## 

### Renting a Bike by Nathan Ainsley (18028669)

|  |
| --- |
| **Use Case: Renting a bike** |
| Owner: Renting a bike |
| **Pre-Conditions** |
| * Bike is available * Rental order is received |
| **Post-Conditions** |
| * Bike should be in the hands of the customer * Pete (rental manager) has deposit and rental record |
| **Primary Path** |
| 1. Rental order is passed to Pete from admin staff, order is then logged into the system 2. Bike engineers get notified of rental order and do an inspection of the bike to make sure that it is safe for use. 3. Customer is informed that the bike is ready to be rented |
| **Alternate Path** |
| 1. Customer decided that they don’t want to rent anymore   A – Rental is then cancelled, and record is marked as cancelled, Deposit is then kept by ray.   1. Bike isn’t safe so rental to be rented.   A – Customer is informed that bike isn’t safe to be rented  B – Customer is then asked if they would like to rent a different bike that is similar  C – If customer decides not to rent a different bike then Deposit is returned and order is marked as cancelled and deposit was returned on the system. |
| **Notes** |
| https://i.gyazo.com/b5122018060c331d6343d51a4d621925.png |

## Top Down ERD



## RDA with Bottom up ERD

# 

**UNF**

Bike ID

Bike Model

Bike Manu ID

Bike Manu Name

Bike Manu address

Bike Manu postcode

Bike Manu telephone

Date bought

Price bought at

Bike Classification

Bike Size

Bike Faults

Bike Faults Date

Bike Fix

Bike Fix Date

Bike Service

Bike Service date

Rental Date

Rental Time Out

Rental time Due

Rental Actual time returned

Client name

Client address

Client postcode

Client telephone

Client paid

**1 NF**

Bike ID

Bike Model

Bike Manu ID

Bike Manu Name

Bike Manu address

Bike Manu postcode

Bike Manu telephone

Date bought

Price bought at

Bike Classification

Bike Size

Bike Faults

Bike Faults Date

Bike Fix

Bike Fix Date

Bike Service

Bike Service date

Rental Date

Rental Time Out

Rental time Due

Rental Actual time returned

Client name

Client address

Client postcode

Client telephone

Client paid

# 

**2 NF**

Bike ID

Bike Model

Bike Manu ID

Bike Manu Name

Bike Manu address

Bike Manu postcode

Bike Manu telephone

Date bought

Price bought at

Bike Classification

Bike Size

Bike Faults

Bike Faults Date

Bike Fix

Bike Fix Date

Bike Service

Bike Service date

Rental Date

Rental Time Out

Rental time Due

Rental Actual time returned

Client name

Client address

Client postcode

Client telephone

Client paid

# 

**3NF**

**Bike ID**

Bike Model

Bike Manu ID

Date bought

Price bought at

Bike Classification

Bike Size

**Bike Manu ID**

Bike Manu Name

Bike Manu address

Bike Manu postcode

Bike Manu telephone

**Bike Maintenance ID**

**Bike ID**

Bike Faults

Bike Faults Date

Bike Fix

Bike Fix Date

**Bike Service ID**

**Bike ID**

Bike Service

Bike Service date

**Rental ID**

**Bike ID**

Rental Date

Rental Time Out

Rental time Due

Rental Actual time returned

**Client ID**

Client paid

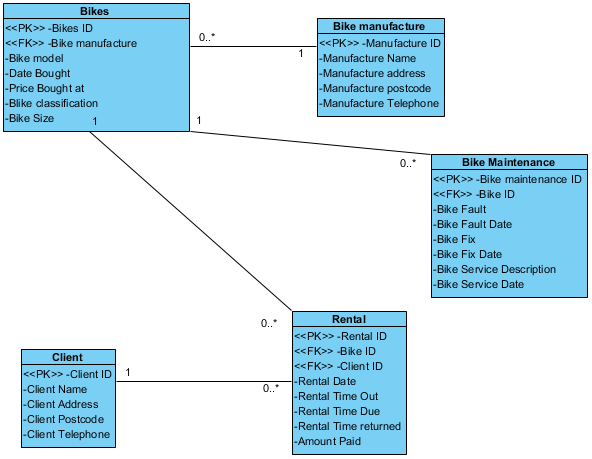
**Client ID**

Client name

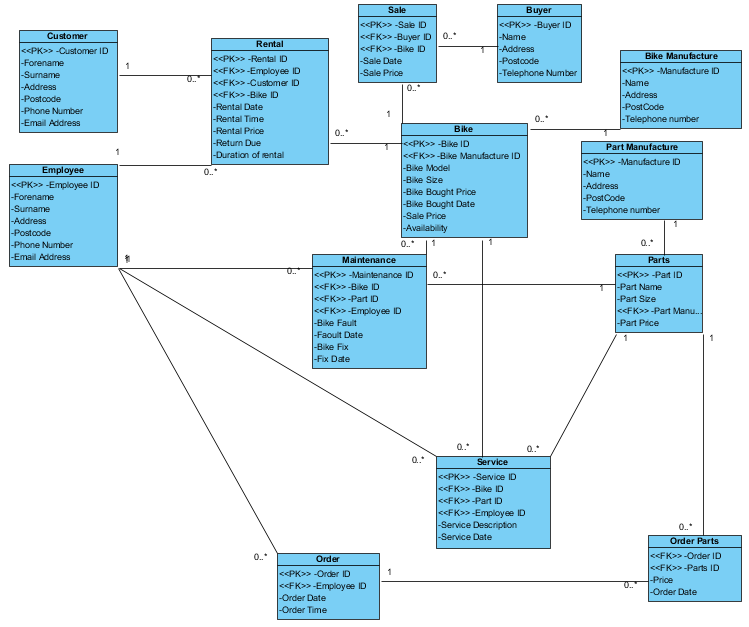
Client address

Client postcode

Client telephone



## Finalised group ERD



## Final ERD Summary

When creating the finalised ERD it was important to consider both the Top down and Down up perspectives of the systems ERD as both on their own don’t give much of an idea of how the system would be represented in a whole but together can show how the system might come together. This is because despite coming from the same data the Top down and Down up ERD’s end up completely different due to the way they are generated from the original date thus by combining the two and using the best bits of both we could then generate the final ERD which considered both perspectives. When creating the Final ERD we needed to keep in mind that the system will have to be simple to use and be able to speed up the rate at which an order can be processed and also the efficiency of handling records. With this in mind, we made sure that each table had an appropriate name and all the fields within the table had sensible names. The system will have to be simple to flow through which is why there aren’t many complicated links in the ERD and most of them are simple 1 to many between two tables although there are a couple where there are more than one links to a certain table with a few tables connecting to as many as 4 other tables.

From creating the Final ERD we learnt how to tackle the problem from different angles to get a better idea on what the best final outcome would be for the system. By doing the TOP DOWN and the DOWN UP ERD’s we were able to more clearly visualise what it is that we needed to do to create the best possible ERD that would be needed. We also used Use case diagrams to think about how a person might go about doing a task and the steps involved with that. Using that knowledge we were able to include those kinds of tasks into the final ERD by making it so that tables that would be used in those tasks where connected and the job would be a lot easier to complete for the person doing said task.

## Conclusion

To conclude, from the diagrams that we have made and the specifications, we have learnt the importance of such ideas/diagrams. They are vital to making and developing a steady, reliable and efficient system.

# Part 3

## Introduction

In this report we will be documenting our implementation of previous ideas in ORACLE SQL developer. Using our Amended ERD, we created a data dictionary table so that we could then create their respective tables in SQL Developer. Once that was completed, we entered three records into each table so that when the queries/management reports had been developed we actually had some data in them to make sure that the reports were working as intended. After careful consideration we decided to change the queries we discussed in part one and develop different, but still similar queries that could get us a better mark by using characteristics that are quite complex and give a more in depth report.

Included in this report is all necessary documents, i.e.; the data dictionary table, our amended ERD, screenshots of our query scripts, screenshots of the output to those queries, and screenshots of the console output when our master SQL file that creates the tables and inserts data is ran. All of this is a long side a brief description of what all of us have learnt during this part of the assignment.

## What we have learnt

### Nathan Ainsley:

Before undertaking this task of creating this database I had quite a bit of experience with SQL language from using MySQL in python to create a database for A level coursework. However creating the database and searches in just SQL proved a bit more of a challenge than expected because I did not have a language that the SQL was within to do certain searches. Despite this, I managed to work out how to do some more complex queries than expected, one included using a text box for the search field to allow the user to choose what kind of bike they are going to search. For the second query included grouping multiple records together to get the record with the closest date to the system date thus getting the most recent record.

The method of defining the foreign keys was also different to how I did it before and was much easier than when I did it in Python, this was also the case with adding constraints to the tables. However, some constraints I tried to enter did not work as I was used to doing them one way, which seems to be specific to MySQL and did not work in Oracle as it was returning violation errors whenever we tried to enter data into the table. The constraints that didn’t work were format checks where we checked to see if an input was in the correct format by checking each character in the string to make sure it was what was wanted, we ended up removing this and just having the basic checks unfortunately.

### Joe Davies:

Before we started part three, and began implementing our ideas from part one I had very little experience using the SQL language and kind of pushed it all off as being boring. This section of the assignment has taught me that it is actually a massive part of how a business works, and is the bridge between business and computing, which sparked a bigger interest in it for me.

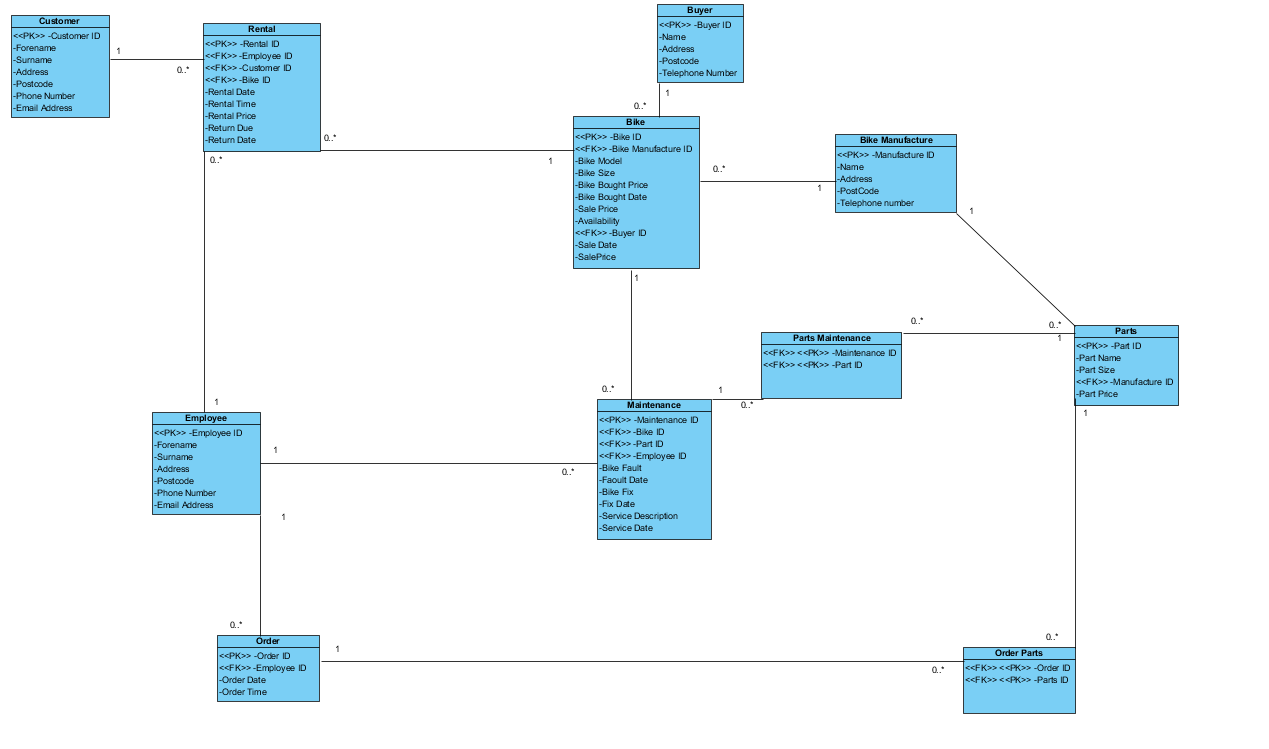
From the lectures and labs I learnt the general syntax for creating SQL statements, which I then used to create the necessary tables, insert data into them, and then run queries/management reports on them. Because of those lessons I was able to help develop queries that include many different kinds of operators and ideas that allow for a more in depth query or management reports. These include; Join of two tables, join of three tables, comparison, arithmetic, and logical operators, use of sys date, and use of group by functions.

### James Brito Alvarado:

When creating this database I learned many new techniques. First, we learnt how to create tables using the ‘CREATE’, ‘DROP’ and ‘INSERT’ commands. When doing this I would then input whether it is a ‘VARCHAR’ or a ‘NUMBER’, then stating whether it is a primary key, foreign key or ‘NOT NULL’ we then learnt how to put data in the tables that we created by inserting data values into the system. We then created queries that allowed us to get certain information by using the ‘SELECT’ and ‘FROM’ commands.

In addition, I learned the design process of a database and the fact that the ERD had to be perfect with correct links, also once that was created a data dictionary was needed. To conclude I had past experience with making databases by using Microsoft Access however this is completely different to ORACLE SQL developer as it is created using code allowing me to learn a new way of creating a database.

## Ammended ERD



## Data Dictionary

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Customer** | **Key Type** | **FK Table** | **FK Column** | **Data Type** | **Length (Characters)** | **Constraint** |
| Customer\_ID | PK | / | / | Integer | 4 | N/a |
| Forename | / | / | / | String | 10 | N/a |
| Surname | / | / | / | String | 25 | N/a |
| Address | / | / | / | String | 40 | Must Contain an integer at the beginning followed by a string |
| Postcode | / | / | / | String | 7 | Must be in capitals and show not be any longer than 7 |
| Phone\_number | / | / | / | Integer | 12 | Must have exactly 12 numbers |
| Email\_Address | / | / | / | String | 40 | Must contain @ symbol followed by an email domain |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Rental** | **Key Type** | **FK Table** | **FK Column** | **Data Type** | **Length** | **Constraint** |
| Rental\_ID | PK | / |  | Integer | 4 | N/a |
| Employee\_ID | FK | Employee | Employee ID | Integer | 4 | N/a |
| Customer\_ID | FK | Customer | Customer ID | Integer | 4 | N/a |
| Bike\_ID | FK | Bike | Bike ID | Integer | 4 | N/a |
| Rental\_Date | / | / | / | Integer | 8 | Must be in this format \_\_/\_\_/\_\_ |
| Rental\_Time | / | / | / | Integer | 5 | Need to be in a digital time \_\_:\_\_ |
| Rental\_Price | / | / | / | Integer | 10,2 | Must contain £ followed by the price |
| Return\_Due | / | / | / | Integer | 8 | Must be in this format \_\_/\_\_/\_\_ |
| Return\_Date | / | / | / | Integer | 8 | Must be in this format \_\_/\_\_/\_\_ |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Bike** | **Key Type** | **FK Table** | **FK Column** | **Data Type** | **Length** | **Constraint** |
| Bike\_ID | PK | / | / | Integer | 4 | N/a |
| Bike\_Manufacturer\_ ID | FK | Bike Manufacturer | Manufacturer ID | Integer | 4 | N/a |
| Bike\_Model | / | / | / | String | 15 | A serial number is required for the bike |
| Bike\_Size | / | / | / | String | 10 | A string with what size the bike is required |
| Bike\_Bought\_Price | / | / | / | Integer | 10,2 | Should be in the format of £\_\_\_\_\_\_\_\_ |
| Bike\_Bought\_Date | / | / | / | String | 8 | Should be in the format of \_\_/\_\_/\_\_ |
| Sale\_Price | / | / | / | Integer | 10,2 | Should be in the format of £\_\_\_\_\_\_\_\_ |
| Availability | / | / | / | Integer | 3 | YES or NO |
| Buyer\_ID | FK | Buyer | Buyer ID | Integer | 4 | N/a |
| Sale\_Date | / | / | / | Integer | 8 | Should be in the format of \_\_/\_\_/\_\_ |
| Sale\_Price | / | / | / | Integer | 6 | Should be in the format of £\_\_\_\_\_\_\_\_ |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Buyer** | **Key Type** | **FK Table** | **FK Column** | **Data Type** | **Length** | **Constraint** |
| Buyer\_ID | PK | / | / | Integer | 4 | N/a |
| Name | / | / | / | String | 30 | N/a |
| Address | / | / | / | String | 40 | Must Contain an integer at the beginning followed by a string |
| Postcode | / | / | / | String | 8 | Must be in capitals and show not be any longer than 7 |
| Telephone\_Number | / | / | / | Integer | 12 | Must be in  (\_\_)\_\_\_\_\_\_\_\_\_ |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Bike manufacture** | **Key Type** | **FK Table** | **FK Column** | **Data Type** | **Length** | **Constraint** |
| Manufacture\_ID | PK | / | / | Integer | 4 | N/a |
| Name | / | / | / | String | 30 | N/a |
| Address | / | / | / | String | 40 | Must Contain an integer at the beginning followed by a string |
| Postcode | / | / | / | String | 8 | Must be in capitals and show not be any longer than 7 |
| Telephone\_Number | / | / | / | Integer | 12 | Must be in  (\_\_)\_\_\_\_\_\_\_\_\_\_  Format |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Parts** | **Key Type** | **FK Table** | **FK Column** | **Data Type** | **Length** | **Constraint** |
| Part\_ID | PK | / | / | Integer | 4 | N/a |
| Part\_Name | / | / | / | String | 30 | N/a |
| Part\_Size | / | / | / | Integer | 10 | N/a |
| Manufacturer\_ID | FK | Manufacture | / | Integer | 4 | N/a |
| Part\_Price | / | / | / | Integer | 10,2 | Must be £\_\_\_\_\_\_  Format |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Part Maintenance** | **Key Type** | **FK Table** | **FK Column** | **Data Type** | **Length** | **Constraint** |
| Maintenance\_ID | FK | Maintenance | Maintenance ID | Integer | 4 | N/a |
| Part\_ID | FK | Part | Part | Integer | 4 | N/a |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Maintenance** | **Key Type** | **FK Table** | **FK Column** | **Data Type** | **Length** | **Constraint** |
| Maintenance\_ID | PK | / | / | Integer | 4 | N/a |
| Bike\_ID | FK | / | Bike ID | Integer | 4 | N/a |
| Part\_ID | FK | / | Part ID | Integer | 4 | N/a |
| Employee\_ID | FK | / | Employee ID | Integer | 4 | N/a |
| Bike\_Fault | / | / | / | String | 15 | N/a |
| Fault\_Date | / | / | / | Integer | 8 | Must be in this format \_\_/\_\_/\_\_ |
| Bike\_Fixed | / | / | / | String | 3 | Either YES or NO |
| Service\_Description | / | / | / | String | 40 | N/a |
| Fix\_Date | / | / | / | Integer | 8 | Must be in this format \_\_/\_\_/\_\_ |
| Service\_Date | / | / | / | Integer | 8 | Must be in this format \_\_/\_\_/\_\_ |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Employee** | **Key Type** | **FK Table** | **FK Column** | **Data Type** | **Length** | **Constraint** |
| Employee\_ID | PK | / | / | Integer | 4 | N/a |
| Forename | / | / | / | String | 10 | N/a |
| Surname | / | / | / | String | 25 | N/a |
| Address | / | / | / | String | 40 | Must Contain an integer at the beginning followed by a string |
| Postcode | / | / | / | Integer | 8 | Must be in capitals and show not be any longer than 7 |
| Phone\_Number | / | / | / | Integer | 12 | Must be in  (\_\_)\_\_\_\_\_\_\_\_\_\_  Format |
| Email\_Address | / | / | / | String | 30 | Must contain @ symbol followed by an email domain |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Order** | **Key Type** | **FK Table** | **FK Column** | **Data Type** | **Length** | **Constraint** |
| Order\_ID | PK | / | / | Integer | 4 | N/a |
| Employee\_ID | FK | Employee | Employee\_ID | Integer | 4 | N/a |
| Order\_Date | / | / | / | Integer | 8 | Must be in this format \_\_/\_\_/\_\_ |
| Order\_Time | / | / | / | Integer | 9 | Must be in this format \_\_:\_\_:\_\_ |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Order Parts** | **Key Type** | **FK Table** | **FK Column** | **Data Type** | **Length** | **Constraint** |
| Order\_ID | FK | Order | Order\_ID | Integer | 4 | N/a |
| Parts\_ID | FK | Parts | Part\_ID | Integer | 4 | N/a |

## SQL Developer Query Screenshots

Along with the submission of this word document, we have also submitted the SQL file, which contains; all drop statements, all table creation statements, all data insert statements, and all four queries. The below figures show screenshots of the individual queries that were ran, and their respective output.

### Query/Report 1.

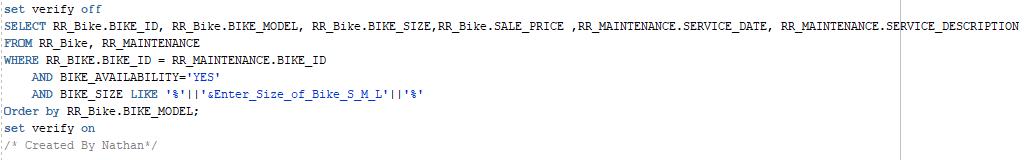
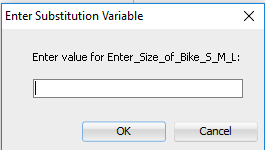
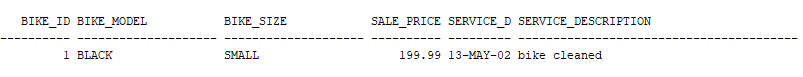
Figure 3 shows the query ran that allows the user to search for an available bike in the size they enter.

Figure 3.1 shows the prompt the user is given, and 3.2 shows the output when “small” is entered.

*Figure 3.*

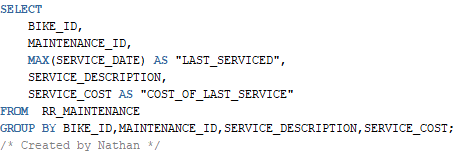


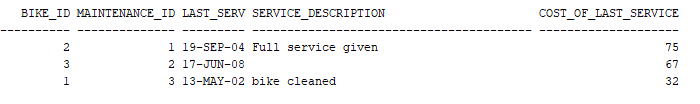
*Figure 3.1.*

*Figure 3.2.*

### Query/Report 2.

Figure 4 shows the query ran that gives the user full service descriptions of current bikes and the cost of the last service given. Figure 4.1 shows the output when this script is ran.

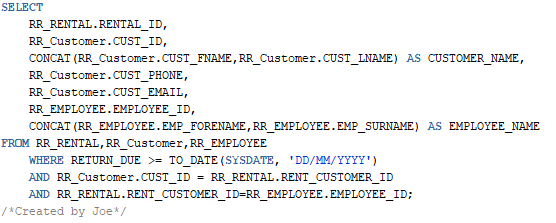


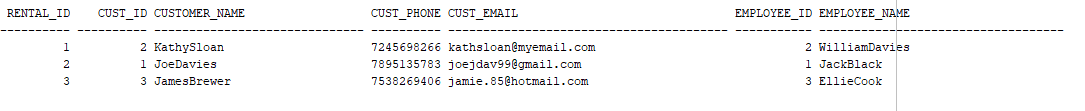
*Figure 4.0.*

*Figure 4.1.*

### Query/Report 3.

Figure 5 show the query ran that gives the user all bikes that are due back and the relevant information regarding customers/employees with it. Figure 5.1 shows the output when the script is ran.

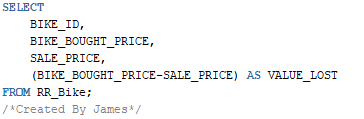


*Figure 5.0.*

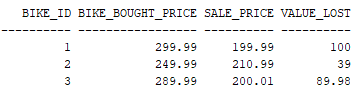
*Figure 5.1.*

### Query/Report 4.

Figure 6.0 shows the query ran that gives the user a report on all bikes that have been sold and includes the value lost in the report, by calculating the price it was bought for minus the price it was sold for. Figure 6.1 shows the output when the script is ran.



*Figure 6.0.*

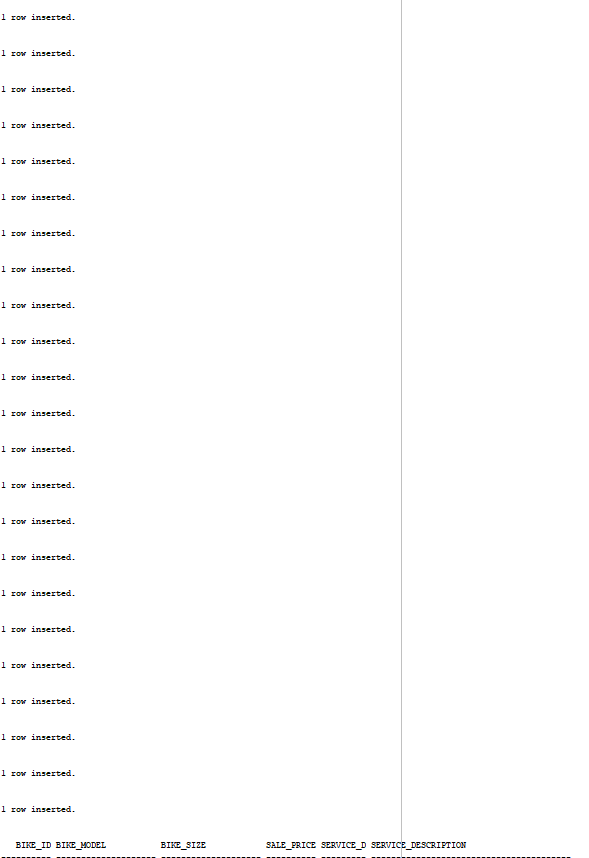
**

*Figure 6.1.*

## SQL Developer script output.

Below shows the output in SQL Developer when our SQL script is ran, which includes all drops, table creations, and insert statements. Copy of the script is in the appendix





# C:\Users\Joe\Documents\UNI STUFF\Year1\IS\FinalPart3\Script3.PNG

# What people have learnt total

James: During the making of the system I oversaw keeping the log book up to date, having no previous experience with creating a database I entered this task with an open mind. In the third part of the assignment I learnt from other group members how to code in sql and create the database. I then used this knowledge to create tables for the system.

Joe: Having never done any sql before entering this task I had no idea what I was going to do as I had never done this language before. However, during part 3 of the project I managed to learn how to code in sql and I am now confident in the language, I oversaw inputting all the data into the system which helped me to get use to the syntax. I also learnt exactly how both computer systems and businesses work together with regards to a database.

Nathan: Having previously created a database system using sql I knew the syntax of the language quite well but the system I created before was nowhere near the scale of this one which was daunting to say the least. However, once we got the ERD created and knew what we where going to create the final system.

Oli: Having previously created a sql system I knew how to create a system, but I hadn’t done as much paperwork for my previous system as I had to do for this system which was a big task.

# Appendix

## References

(Analysis Reports, Oracle Hospitality Reporting and Analytics User Guide):

Oracle Help Center, 2018, https://docs.oracle.com/cd/E71871\_01/doc.90/e72023/c\_reports\_analysis\_examples.htm#RAUSE-AnalysisReportExamples-0D11C9A0

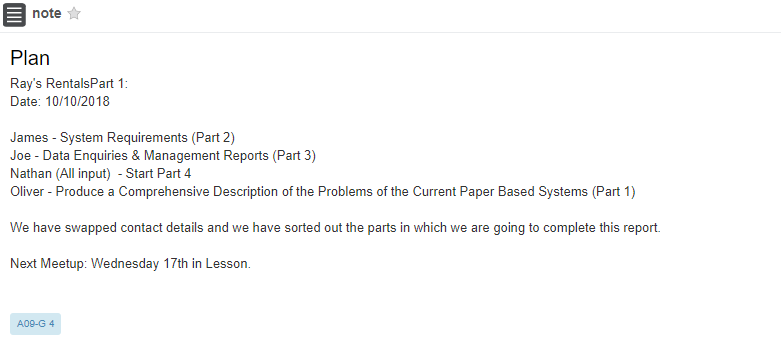
(Exception Report Template, date: unknown):

http://viddr.me/exception-report-template/exception-report-template-exception-report-road-safety-audit-exception-report-template/

## Piazza Posts

### Part 1:

Meet up one:

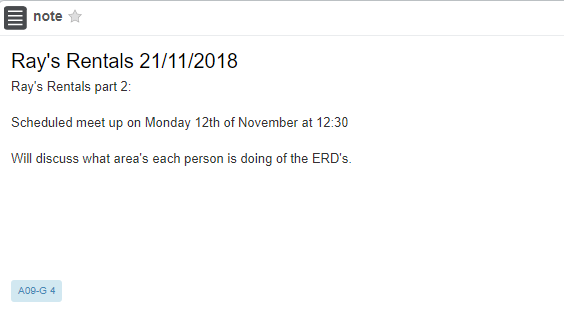


Meet up two:

### 

### Part 2:

Meeting one:



Meeting two:

### 

Meeting three:

### Meeting four:

### 

Meeting five:

## 

### Part 3:

Meetup 1

#### 

Meetup 2

#### 

Meetup 3

#### 

### Part 4:

Meetup 1

#### 

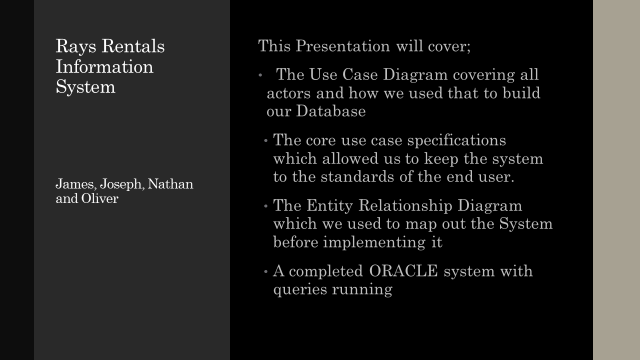
Meetup 2:

#### 

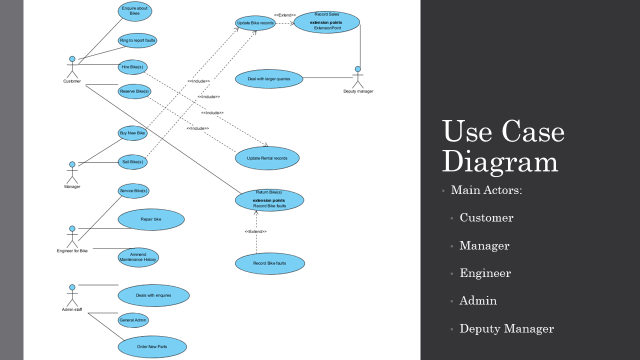
Meetup 3:

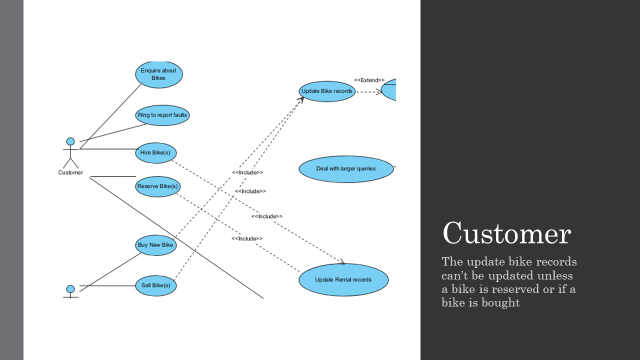
#### 

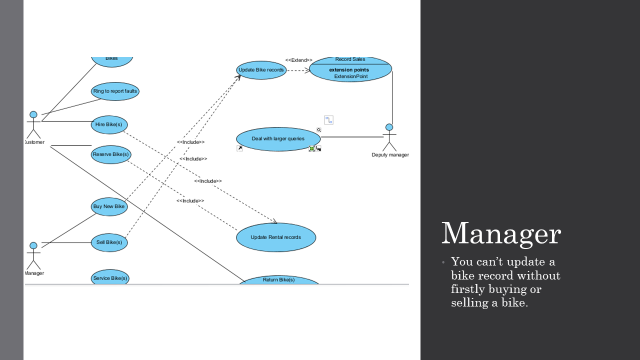
# Presentation

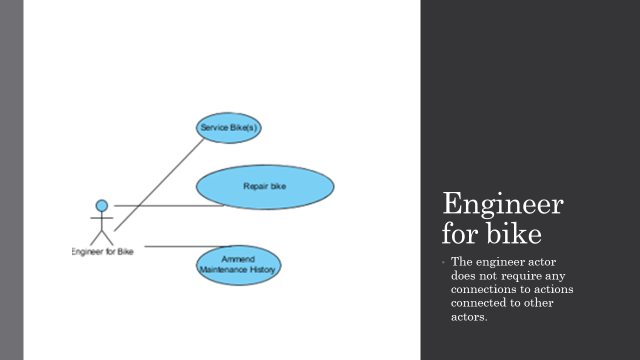


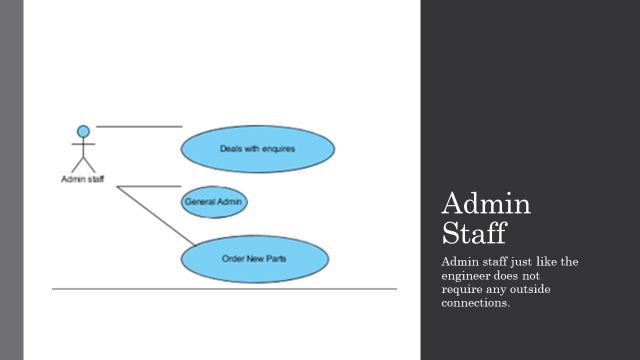
# 

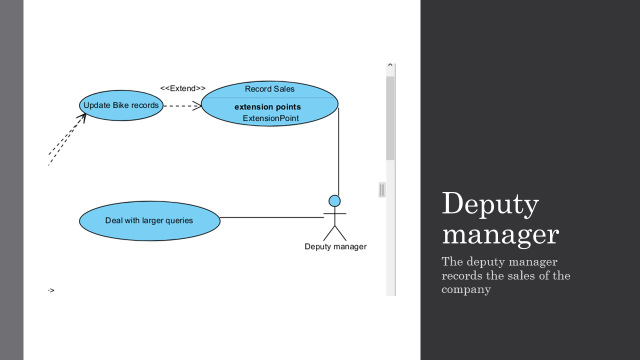






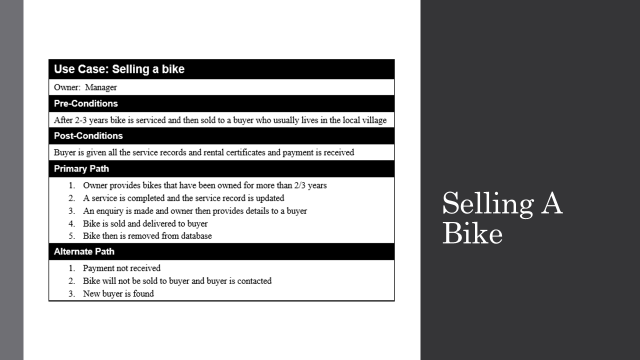


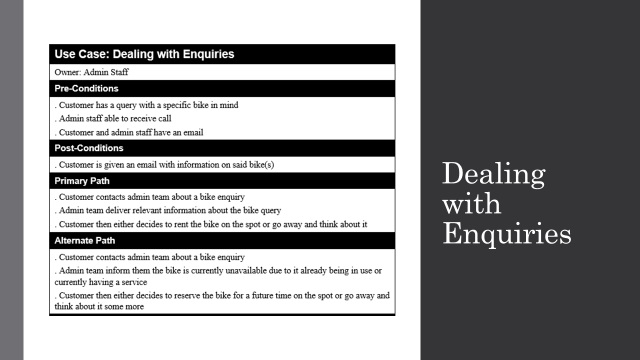


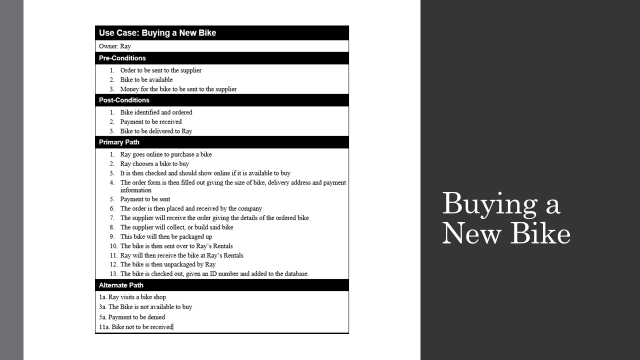


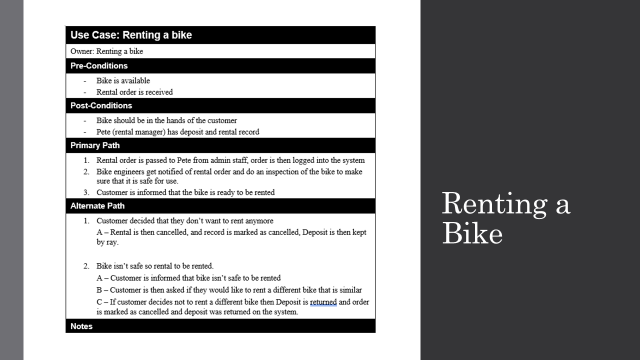


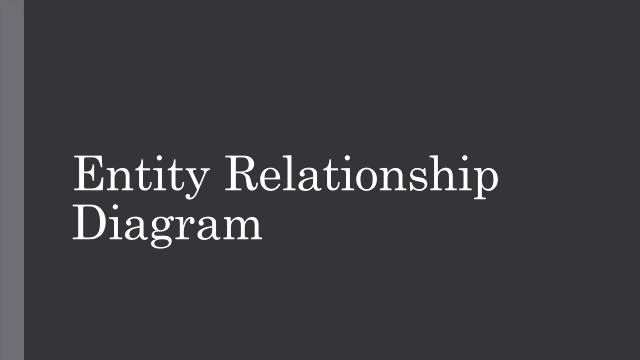


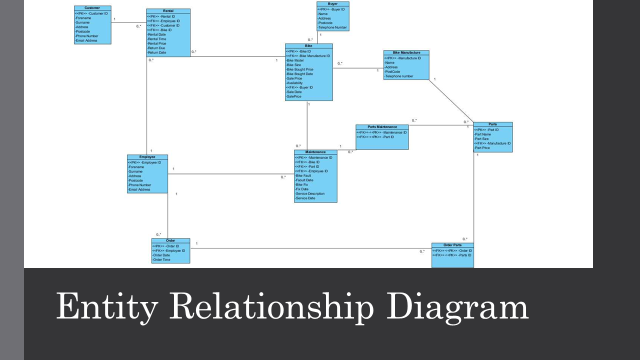




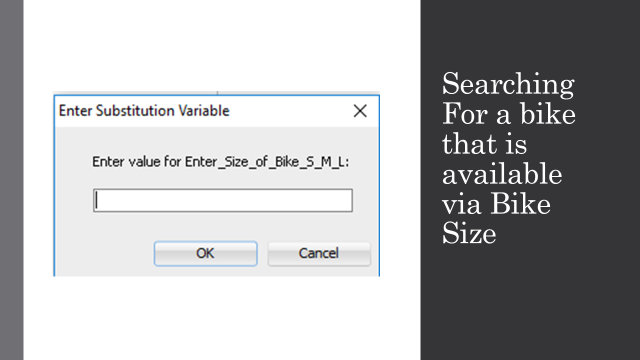




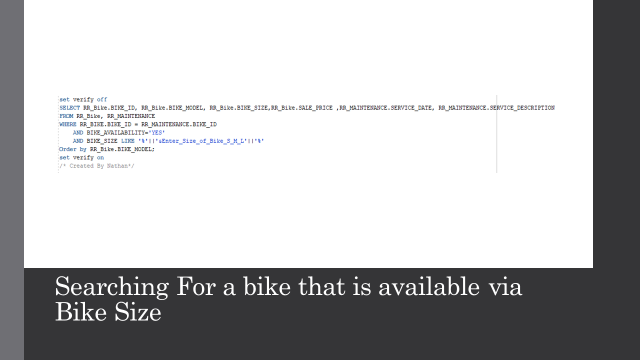


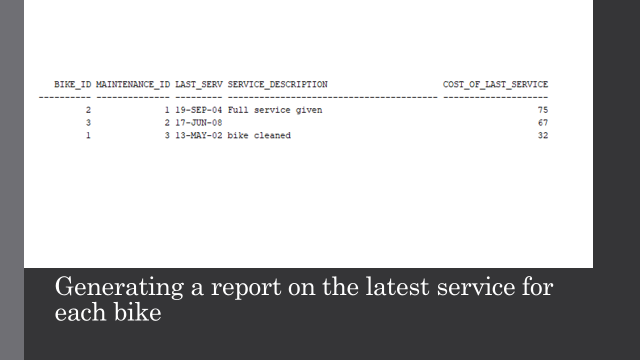




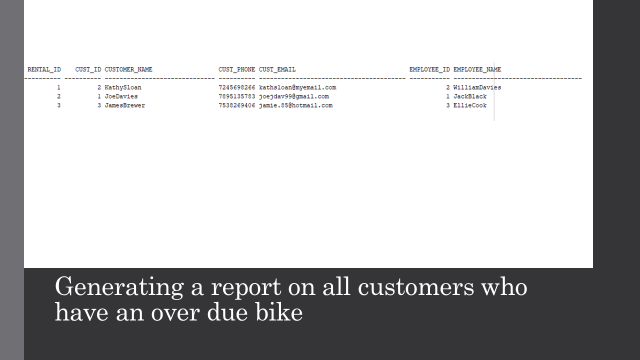


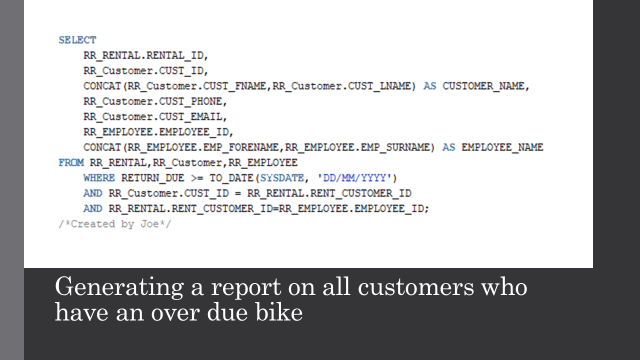


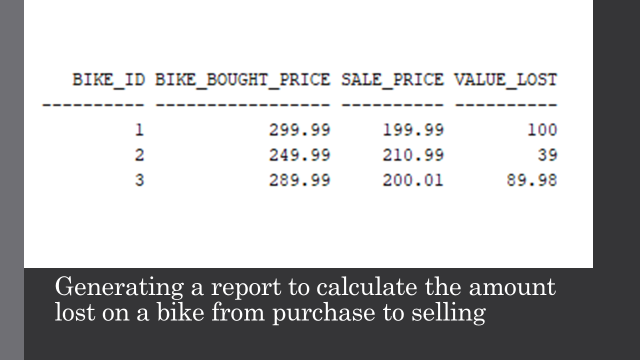


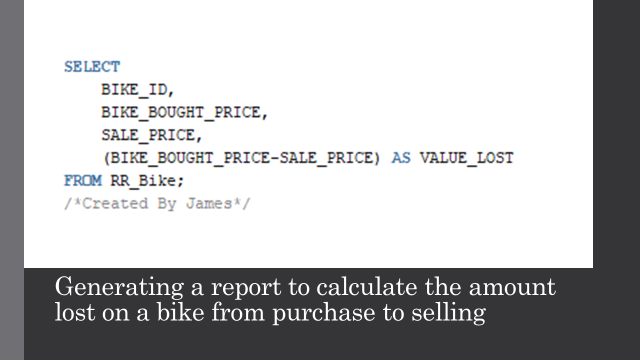
















# Copy of script file

DROP TABLE RR\_PARTS\_MAINTENANCE;

DROP TABLE RR\_MAINTENANCE;

DROP TABLE RR\_ORDER\_PARTS;

DROP TABLE RR\_Order;

DROP TABLE RR\_Rental;

DROP TABLE RR\_Bike;

DROP TABLE RR\_Parts;

DROP TABLE RR\_Buyer;

DROP TABLE RR\_Manufacturer;

DROP TABLE RR\_Employee;

DROP TABLE RR\_Customer;

CREATE TABLE RR\_Customer

(

CUST\_ID NUMBER(4) PRIMARY KEY,

CUST\_FNAME VARCHAR(15) NOT NULL,

CUST\_LNAME VARCHAR(15) NOT NULL,

CUST\_ADDRESS VARCHAR(20) NOT NULL,

CUST\_POSTCODE VARCHAR(8) NOT NULL,

CUST\_PHONE NUMBER(11) NOT NULL,

CUST\_EMAIL VARCHAR(40) NOT NULL

/\* Created by Nathan \*/

);

CREATE TABLE RR\_Employee

(

EMPLOYEE\_ID NUMBER(4) PRIMARY KEY,

EMP\_FORENAME VARCHAR(15) NOT NULL,

EMP\_SURNAME VARCHAR(20) NOT NULL,

EMP\_ADDRESS VARCHAR(40) NOT NULL,

EMP\_POSTCODE VARCHAR(8) NOT NULL,

EMP\_PHONE NUMBER(11) NOT NULL,

EMP\_EMAIL VARCHAR(40) NOT NULL

/\* Made by Oliver Bevington \*/

);

CREATE TABLE RR\_Manufacturer

(

MANUFACTURER\_ID NUMBER(4) PRIMARY KEY,

MANUFACTURER\_NAME VARCHAR(30) NOT NULL,

MANUFACTURER\_ADDRESS VARCHAR(40) NOT NULL,

MANUFACTURER\_POSTCODE VARCHAR(8) NOT NULL,

MANUFACTURER\_PHONE NUMBER(12) NOT NULL

/\* Made by Joe Davies \*/

);

CREATE TABLE RR\_Buyer

(

BUYER\_ID NUMBER(4) PRIMARY KEY,

BUYER\_NAME VARCHAR(30) NOT NULL,

BUYER\_ADDRESS VARCHAR(40) NOT NULL,

BUYER\_POSTCODE VARCHAR(8) NOT NULL,

BUYER\_PHONE NUMBER(12) NOT NULL

/\* Made by Joe Davies \*/

);

CREATE TABLE RR\_Parts

(

PART\_ID NUMBER(4) PRIMARY KEY,

PART\_NAME VARCHAR(30) NOT NULL,

PART\_SIZE VARCHAR(10) NOT NULL,

PART\_MANUFACTURER\_ID NUMBER(4) references RR\_Manufacturer(MANUFACTURER\_ID),

PART\_PRICE NUMBER(10,2) NOT NULL,

CONSTRAINT Chk\_Size CHECK (PART\_SIZE = 'SMALL' OR PART\_SIZE = 'MEDIUM' OR PART\_SIZE = 'LARGE')

/\*made by James Brito Alvarado \*/

);

CREATE TABLE RR\_Bike

(

BIKE\_ID NUMBER(4) PRIMARY KEY,

BK\_MANUFACTURER\_ID NUMBER(4) references RR\_Manufacturer(MANUFACTURER\_ID),

BIKE\_MODEL VARCHAR(20) NOT NULL,

BIKE\_SIZE VARCHAR(20) NOT NULL,

BIKE\_BOUGHT\_PRICE NUMBER(10,2) NOT NULL,

BIKE\_BOUGHT\_DATE DATE NOT NULL,

SALE\_PRICE NUMBER(10,2) NOT NULl,

BIKE\_AVAILABILITY VARCHAR(3) NOT NULL,

BK\_BUYER\_ID NUMBER(4) references RR\_Buyer(BUYER\_ID),

SALE\_DATE DATE NOT NULL,

CONSTRAINT Chk\_bkAvailability CHECK (BIKE\_AVAILABILITY = 'YES' OR BIKE\_AVAILABILITY = 'NO'),

CONSTRAINT Chk\_bkSize CHECK (BIKE\_SIZE = 'SMALL' OR BIKE\_SIZE = 'MEDIUM' OR BIKE\_SIZE = 'LARGE')

);

/\*Made by Joe Davies\*/

CREATE TABLE RR\_Rental

(

Rental\_ID NUMBER(4) PRIMARY KEY,

RENT\_EMPLOYEE\_ID NUMBER(4) REFERENCES RR\_Employee (EMPLOYEE\_ID),

RENT\_CUSTOMER\_ID NUMBER(4) REFERENCES RR\_Customer (CUST\_ID),

BIKE\_ID NUMBER(4) REFERENCES RR\_Bike (BIKE\_ID),

RENTAL\_DATE DATE NOT NULL,

RENTAL\_TIME VARCHAR(9) NOT NULL,

RENTAL\_PRICE NUMBER(10,2) NOT NULL,

RETURN\_DUE DATE NOT NULL,

RETURN\_DATE DATE

/\* Created by Nathan \*/

);

CREATE TABLE RR\_Order

(

ORDER\_ID NUMBER(4) PRIMARY KEY,

EMPLOYEE\_ID NUMBER(4) references RR\_Employee(EMPLOYEE\_ID),

ORDER\_DATE DATE NOT NULL,

ORDER\_TIME VARCHAR(9) NOT NULL

/\* Made by Oliver Bevington \*/

);

CREATE TABLE RR\_ORDER\_PARTS

(

LNK\_ORDER\_ID NUMBER(4) references RR\_Order(ORDER\_ID),

LNK\_ORDER\_PART\_ID NUMBER(4) references RR\_Parts(PART\_ID),

PRIMARY KEY (LNK\_ORDER\_ID,LNK\_ORDER\_PART\_ID)

/\*Made by Oliver Bevington\*/

);

CREATE TABLE RR\_MAINTENANCE

(

MAINTENANCE\_ID NUMBER(4) PRIMARY KEY,

BIKE\_ID NUMBER(4) references RR\_Bike(BIKE\_ID),

PART\_ID NUMBER(4) references RR\_Parts(PART\_ID),

EMPLOYEE\_ID NUMBER(4) references RR\_Employee(EMPLOYEE\_ID),

BIKE\_FAULT VARCHAR(15) NOT NULL,

FAULT\_DATE DATE NOT NULL,

BIKE\_FIXED VARCHAR(3),

SERVICE\_DESCRIPTION VARCHAR(40),

FIX\_DATE DATE,

SERVICE\_DATE DATE,

SERVICE\_COST NUMBER(10,2) NOT NULL,

CONSTRAINT CHK\_BIKE\_FIXED CHECK (BIKE\_FIXED = 'YES' OR BIKE\_FIXED = 'NO')

/\*Made by Nathan Ainsley \*/

);

CREATE TABLE RR\_PARTS\_MAINTENANCE

(

LNK\_MAINTENANCE\_ID NUMBER(4) references RR\_Manufacturer(MANUFACTURER\_ID),

LNK\_PART\_ID NUMBER(4) references RR\_Parts(PART\_ID),

PRIMARY KEY(LNK\_MAINTENANCE\_ID,LNK\_PART\_ID)

/\* Made by James Brito Alvarado \*/

);

/\*DATA INSERT SECTION\*/

/\*CUSTOMER\*/

INSERT INTO RR\_Customer(CUST\_ID,CUST\_FNAME,CUST\_LNAME,CUST\_ADDRESS,CUST\_POSTCODE,CUST\_PHONE,CUST\_EMAIL)

VALUES(0001,'Joe','Davies','39 Primrose Lane','yo12 7ee',07895135783,'joejdav99@gmail.com');

INSERT INTO RR\_CUSTOMER(CUST\_ID,CUST\_FNAME,CUST\_LNAME,CUST\_ADDRESS,CUST\_POSTCODE,CUST\_PHONE,CUST\_EMAIL)

VALUES(0002,'Kathy','Sloan','565 King Street','bg54 8fr',07245698266,'kathsloan@myemail.com');

INSERT INTO RR\_CUSTOMER(CUST\_ID,CUST\_FNAME,CUST\_LNAME,CUST\_ADDRESS,CUST\_POSTCODE,CUST\_PHONE,CUST\_EMAIL)

VALUES(0003,'James','Brewer','45 Latham Avenue','gh64 0hj',07538269406,'jamie.85@hotmail.com');

/\*EMPLOYEE\*/

INSERT INTO RR\_Employee(EMPLOYEE\_ID,EMP\_FORENAME,EMP\_SURNAME,EMP\_ADDRESS,EMP\_POSTCODE,EMP\_PHONE,EMP\_EMAIL)

VALUES(0001,'Jack','Black','76 Heart Road','sh99 2gn',07455890901,'Jack@hotmail.com');

INSERT INTO RR\_Employee(EMPLOYEE\_ID,EMP\_FORENAME,EMP\_SURNAME,EMP\_ADDRESS,EMP\_POSTCODE,EMP\_PHONE,EMP\_EMAIL)

VALUES(0002,'William','Davies','44 Grim Street','fg56 2gg',07676743590,'okay@gmail.com');

INSERT INTO RR\_Employee(EMPLOYEE\_ID,EMP\_FORENAME,EMP\_SURNAME,EMP\_ADDRESS,EMP\_POSTCODE,EMP\_PHONE,EMP\_EMAIL)

VALUES(0003,'Ellie','Cook','1 Cook Street','ja76 9hg',07895135783,'runninfnames@hotmail.com');

/\*BUYER\*/

INSERT INTO RR\_BUYER(BUYER\_ID,BUYER\_NAME,BUYER\_ADDRESS,BUYER\_POSTCODE,BUYER\_PHONE)

VALUES(0001,'Jasmine Cook','56 Kingsley Road','kj67 9ll',07657613290);

INSERT INTO RR\_BUYER(BUYER\_ID,BUYER\_NAME,BUYER\_ADDRESS,BUYER\_POSTCODE,BUYER\_PHONE)

VALUES(0002,'Sophie Jones','12 Maguire Street','gj22 7uu',076574617834);

INSERT INTO RR\_BUYER(BUYER\_ID,BUYER\_NAME,BUYER\_ADDRESS,BUYER\_POSTCODE,BUYER\_PHONE)

VALUES(0003,'Jack Whitehead','23 Primrose Lane','hj43 2ba',07567891320);

/\*MANUFACTURER\*/

INSERT INTO RR\_MANUFACTURER(MANUFACTURER\_ID,MANUFACTURER\_NAME,MANUFACTURER\_ADDRESS,MANUFACTURER\_POSTCODE,MANUFACTURER\_PHONE)

VALUES(0001,'Jonathan Jones','19 Hillview Road','wa67 0hh',07134894358);

INSERT INTO RR\_Manufacturer(MANUFACTURER\_ID,MANUFACTURER\_NAME,MANUFACTURER\_ADDRESS,MANUFACTURER\_POSTCODE,MANUFACTURER\_PHONE)

VALUES(0002,'Alex Usher','12 Boy Road','fg91 5oj',07472573890);

INSERT INTO RR\_MANUFACTURER(MANUFACTURER\_ID,MANUFACTURER\_NAME,MANUFACTURER\_ADDRESS,MANUFACTURER\_POSTCODE,MANUFACTURER\_PHONE)

VALUES(0003,'Michelle Green','67 Bored Lane','dp73 4un',07556989231);

/\*BIKE\*/

--INSERT INTO RR\_Bike(BIKE\_ID, BK\_MANUFACTURER\_ID, BIKE\_MODEL, BIKE\_SIZE, BIKE\_BOUGHT\_PRICE, BIKE\_BOUGHT\_DATE, SALE\_PRICE, BIKE\_AVAILABILITY, BK\_BUYER\_ID, SALE\_DATE),

INSERT INTO RR\_Bike VALUES(0001,0001,'BLACK','SMALL','299.99',TO\_DATE('2004/04/12','yyyy/mm/dd'),'199.99','YES',0002,TO\_DATE('2005/09/09','yyyy/mm/dd'));

INSERT INTO RR\_Bike VALUES(0002,0003,'ORANGE','LARGE','249.99',TO\_DATE('2007/03/11','yyyy/mm/dd'),'210.99','NO',0001,TO\_DATE('2004/08/11','yyyy/mm/dd'));

INSERT INTO RR\_Bike VALUES(0003,0001,'ORANGE','MEDIUM','289.99',TO\_DATE('2008/02/12','yyyy/mm/dd'),'200.01','YES',0003,TO\_DATE('2002/04/12','yyyy/mm/dd'));

/\*PARTS\*/

INSERT INTO RR\_PARTS VALUES(0001,'SEAT','SMALL',0002,'30');

INSERT INTO RR\_PARTS VALUES(0002,'GEAR','LARGE',0001,'70');

INSERT INTO RR\_PARTS VALUES(0003,'CHAIN','MEDIUM',0003,'35');

/\*MAINTENANCE\*/

--MAINTENANCE\_ID BIKE\_ID PART\_ID EMPLOYEE\_ID BIKE\_FAULT FAULT\_DATE BIKE\_FIXED SERVICE\_DESCRIPTION FIX\_DATE SERVICE\_DATE SERVICE\_COST

INSERT INTO RR\_MAINTENANCE VALUES(0001,0002,0003,0001,'Broken chain', TO\_DATE('2004/09/12','yyyy/mm/dd'),'YES','Full service given',TO\_DATE('2004/09/18','yyyy/mm/dd'),TO\_DATE('2004/09/19','yyyy/mm/dd'),'75');

INSERT INTO RR\_MAINTENANCE VALUES(0002,0003,0002,0001,'Deflated tires', TO\_DATE('2008/06/12','yyyy/mm/dd'),'YES','',TO\_DATE('2008/06/16','yyyy/mm/dd'),TO\_DATE('2008/06/17','yyyy/mm/dd'),'67');

INSERT INTO RR\_MAINTENANCE VALUES(0003,0001,0001,0002,'Broken chain', TO\_DATE('2002/05/12','yyyy/mm/dd'),'YES','bike cleaned',TO\_DATE('2002/05/12','yyyy/mm/dd'),TO\_DATE('2002/05/13','yyyy/mm/dd'),'32');

/\*PARTS MAINTENANCE\*/

INSERT INTO RR\_PARTS\_MAINTENANCE VALUES(0001,0003);

INSERT INTO RR\_PARTS\_MAINTENANCE VALUES(0002,0002);

INSERT INTO RR\_PARTS\_MAINTENANCE VALUES(0003,0001);

/\*RENTAL\*/

INSERT INTO RR\_RENTAL VALUES(0001,0001,0002,0003,TO\_DATE('2004/09/12','yyyy/mm/dd'),TO\_DATE('13:39:01','HH24/MI/SS'),15.99,TO\_DATE('2004/09/17','yyyy/mm/dd'),TO\_DATE('2004/09/18','yyyy/mm/dd'));

INSERT INTO RR\_RENTAL VALUES(0002,0001,0001,0001,TO\_DATE('2006/06/02','yyyy/mm/dd'),TO\_DATE('22:54:45','HH24/MI/SS'),18.99,TO\_DATE('2006/06/12','yyyy/mm/dd'),TO\_DATE('2006/06/10','yyyy/mm/dd'));

INSERT INTO RR\_RENTAL VALUES(0003,0003,0003,0002,TO\_DATE('2010/03/23','yyyy/mm/dd'),TO\_DATE('07:12:03','HH24/MI/SS'),12.99,TO\_DATE('2010/03/27','yyyy/mm/dd'),TO\_DATE('2010/03/27','yyyy/mm/dd'));

/\*ORDER\*/

INSERT INTO RR\_Order VALUES(0001,0001,TO\_DATE('2004/09/12','yyyy/mm/dd'),TO\_DATE('13:39:01','HH24/MI/SS'));

INSERT INTO RR\_Order VALUES(0002,0003,TO\_DATE('2007/11/08','yyyy/mm/dd'),TO\_DATE('22:54:45','HH24/MI/SS'));

INSERT INTO RR\_Order VALUES(0003,0002,TO\_DATE('2012/04/17','yyyy/mm/dd'),TO\_DATE('07:12:03','HH24/MI/SS'));

/\*ORDER PARTS\*/

INSERT INTO RR\_ORDER\_PARTS VALUES(0001,0003);

INSERT INTO RR\_ORDER\_PARTS VALUES(0002,0001);

INSERT INTO RR\_ORDER\_PARTS VALUES(0003,0002);

set verify off

SELECT RR\_Bike.BIKE\_ID, RR\_Bike.BIKE\_MODEL, RR\_Bike.BIKE\_SIZE,RR\_Bike.SALE\_PRICE ,RR\_MAINTENANCE.SERVICE\_DATE, RR\_MAINTENANCE.SERVICE\_DESCRIPTION

FROM RR\_Bike, RR\_MAINTENANCE

WHERE RR\_BIKE.BIKE\_ID = RR\_MAINTENANCE.BIKE\_ID

AND BIKE\_AVAILABILITY='YES'

AND BIKE\_SIZE LIKE '%'||'&Enter\_Size\_of\_Bike\_S\_M\_L'||'%'

Order by RR\_Bike.BIKE\_MODEL;

set verify on

/\* Created By Nathan\*/

SELECT

BIKE\_ID,

MAINTENANCE\_ID,

MAX(SERVICE\_DATE) AS "LAST\_SERVICED",

SERVICE\_DESCRIPTION,

SERVICE\_COST AS "COST\_OF\_LAST\_SERVICE"

FROM RR\_MAINTENANCE

GROUP BY BIKE\_ID,MAINTENANCE\_ID,SERVICE\_DESCRIPTION,SERVICE\_COST;

/\* Created by Nathan \*/

SELECT

RR\_RENTAL.RENTAL\_ID,

RR\_Customer.CUST\_ID,

CONCAT(RR\_Customer.CUST\_FNAME,RR\_Customer.CUST\_LNAME) AS CUSTOMER\_NAME,

RR\_Customer.CUST\_PHONE,

RR\_Customer.CUST\_EMAIL,

RR\_EMPLOYEE.EMPLOYEE\_ID,

CONCAT(RR\_EMPLOYEE.EMP\_FORENAME,RR\_EMPLOYEE.EMP\_SURNAME) AS EMPLOYEE\_NAME

FROM RR\_RENTAL,RR\_Customer,RR\_EMPLOYEE

WHERE RETURN\_DUE >= TO\_DATE(SYSDATE, 'DD/MM/YYYY')

AND RR\_Customer.CUST\_ID = RR\_RENTAL.RENT\_CUSTOMER\_ID

AND RR\_RENTAL.RENT\_CUSTOMER\_ID=RR\_EMPLOYEE.EMPLOYEE\_ID;

/\*Created by Joe\*/

SELECT

BIKE\_ID,

BIKE\_BOUGHT\_PRICE,

SALE\_PRICE,

(BIKE\_BOUGHT\_PRICE-SALE\_PRICE) AS VALUE\_LOST

FROM RR\_Bike;

/\*Created By James\*/