Information Systems Project

Amanda McIntosh 09081262

Vamsi Pabbineedi 12019750

Precious Igbinosun 13129589

Information Systems Project

1.0 Contents

1.0 Introduction **4**

Part 1 Management Reports ..... **5 - 20**

2.0 Part One Introduction 6
3.0 Problems ..... 7 - 8
- 3.1 Data Integrity 7
- 3.2 Data Security 7
- 3.3 Data Cost & Storage 7
- 3.4 Availability of Data ..... 7 -8
- 3.5 Data Backup 8
- 3.6 Online Availability 8
- 3.7 Forms 8

4.0 System Requirements 9

5.0 Management Reports ..... 10 - 15
- 5.1 Analysis Report 11
- 5.2 Exception Report ..... 12 - 13
- 5.3 Key Target Report ..... 13 - 14
- 5.4 Ad Hoc Report ..... 14 - 15

6.0 Management Reports that will be of use to Ray's Rental ..... 16 - 19
- 6.1 Monthly Profit and Loss Analysis Report 16
- 6.2 Maintenance History Exception Report ..... 17 - 18
- 6.3 Key Target Report 18
- 6.4 Ad Hoc Report 19

7.0 Part One Conclusion 20

Part 2 Systems Analysis & Design ..... **21 - 57**

8.0 Introduction 22
9.0 Use Case Diagram 23
- 9.1 Commentary 24

10.0 Use Case Specification ..... 25 - 51
- 10.1 Amanda's Use Case Specifications ..... 25 - 36
- 10. 2 Vamsi's Use Case Specifications ..... 37 - 47

10. 3 Precious's Use Case Specifications ..... 48 - 51

11.0 RDA 52- 55
11.1 Bike Record RDA ..... 52 - 53
11.2 Rental Record RDA 54
11.3 RDA ERD 55

12.0 ERD 56
12.1 Commentary 57

13.0 Conclusion 57

Part 2 Database Design & Oracle Implementation ..... 58 – 87

14.0 Introduction 59
15.0 Updated ERD 60
15.1 Commentary 60

16.0 Data Dictionary 61-67

17.0 SQL Documentation 68-84
17.1 Amanda McIntosh's SQL Queries 71-75
17.2 Vamsi Pabbineedi 's SQL Queries 76-81
17.3 Precious Igbinosun's SQL Queries 82-84

18.0 Commentary 85
18.1 Amanda McIntosh 85
18.2 Vamsi Pabbineedi 85
18.3 Precious Igbinosun 85

19.0 Project Conclusion ..... 86 - 87

20.0 References 88

APPENDIX A ..... 89 - 114

APPENDIX B ..... 115 - 122

Information Systems Project

1.0 Introduction

This report has been separated into three main sections: Management Reports, Systems Analysis & Design and Database Design & Oracle Implementation. Each of the three parts has a small introduction describing the sections included.

The main aim of this report is to evaluate Ray's Rental business and to design a computer based information system. This database will be tailored to the needs of Ray's rentals. The first part of this report (Management Reports) evaluates the current system available to Ray's rentals and it's problems. It also discusses possible improvement via the use of Management Reports.

The second part of this report (Systems Analysis & Design) evaluates the data requirements of the system and proposes a desirable database. Use case diagrams and entity relationship diagrams have been illustrated to show the structure of the proposed database.

The third section of this report (Database Design and Oracle Implementation) involves the implementation of the Oracle database, the code created for the database and queries has also been provided in Appendix A.

This project has been a join effort between Amanda, Vamsi and Precious (33.3%). Tim contributed the Ad Hoc reports in the Management Reports.

Information Systems Project

Part One

Management Reports

Information Systems Project

2.0 Part One Introduction

The main objective of part one of the report is to evaluate the problems with the paper based information system of Ray's Rentals.

Section 3.0 will outline all the main flaws of a paper based information system, evaluating the causes for these flaws and providing examples from Ray's Rentals.

Section 4.0 discusses the system requirements for Ray's Rentals. These requirements have been deemed essential due to the analysis of the paper based forms currently being used.

The focus of section 5.0 will be on the relevance and purpose of management information reports. This section will discuss four different types of management reports and provide examples.

In section 6.0, the delineated management reports for Ray's Rentals have been explained with examples provided.

Section 7.0 will conclude this section of the report.

3.0 Problems

Ray's Rentals is a small business, which rents bikes to the local area and relies on the loyalty of returning customers and the local tourist trade for its income. Ray's business currently consists of just one shop, which operates a paper based information system.

3.1 Data Integrity

The main problem with this information system is the integrity of the data. Difficulties will arise when a staff member records details incorrectly, cannot understand the handwriting of another employee or simply forgets to add necessary details. Due to the paper based information system, data can quickly become out of date and useless. This can lead to severe consequence if, for example, financial records or tax rebates are incorrect. It is also a legal requirement under the data protection act for the stored data to be accurate. Some of Ray's problems with data integrity include:

- The 'sample prices of bikes list' is usually out of data and needs editing by hand. This is unprofessional and time consuming.
- Telephone enquiries' noted on pieces of paper could be hard to understand and may lead to misinterpretation.

3.2 Data Security

Security of the stored data is also unnecessarily difficult with this paper based information system. To comply with data protection regulations all records that are stored concerning customer details and financial records must be secure. This security would require Ray to keep all of his paper-based files in a locked cabinet or office, whereas with a computer based information system passwords can easily protect data. The Data Protection Act (1998) states that:

"Appropriate technical and organisational measures shall be taken against unauthorised or unlawful processing of personal data and against accidental loss or destruction of, or damage to, personal data"

An example of Ray's problems with data security would be:

- Reservations can get 'lost' resulting in bad customer relations and subsequently a loss of their custom.

3.3 Data Cost & Storage

The storage space required to store information for two years will carry high overheads for Ray's business. Considering the cost of paper, stationary, storage facilities and storage space and comparing this to the cost of running a computer based information system; the costs are preventable. Although the cost of implementing a computer based information system may be high, the overall saving would be greater.

3.4 Availability of Data

Ray's current information system makes the availability of data extremely limited. Trying to compare records from past years would not only require a vast amount of time but would be near impossible and limited to the two years of stored information. This makes useful activities including reporting and targeted mailings impossible. Problems with reservations and on-going enquiries will also arise, as the information needed is not readily available to the member of staff. As data records are paper based there will be

difficulty when transferring records to a different department. For example, when a fault arises with a rented bike the rental team must pass this information onto the maintenance team. The rental team must make a hard copy and pass this, by hand, to the maintenance team; this could result in the loss of records. With a computer based information system, a report could be processed and sent to the maintenance team. Examples of this limitation in Ray's business include:

- Inadequate enquiry handling caused the loss of business. The current system cannot adequately record and store enquiries, this can result in forgotten enquiries.
- Maintenance records are unclear and can cause delays in needed maintenance work. This is mainly because of the misinterpretation of the handwriting on the service list. Also there is no current system that allows the prioritisation of repair work, this can lead to delays and the bike in question being out of commission longer than necessary.
- Stock is not available due to the lack of a stock recording system. This results in delayed maintenance work and the loss of income from the decommissioned bike.
- Maintenance stock is written-off as there is no sufficient record of the stock available. There is also no record of the required amount of stock in storage. The delivery notes are currently stored in a filing cabinet with no sorting system, this makes finding required data difficult and the cross checking of bike parts and the delivery notes impossible.

3.5 Data Backup

One of the key flaws of a paper-based system is the unavailability of backed up data. Using this current information system; to make a backup Ray must make a hard copy of all of his paper records (i.e. scan / archive) and store these at a safe and secure location (i.e. Fireproof filing cabinet). This would carry high costs and significant labour hours.

3.6 Online Availability

The current information system limits the potential of online availability. Enquiry and reservation services are unavailable online due to the information system being paper based. The implementation of a computerised information system would make these features available and information supplied online can automatically update.

3.7 Forms

The design and layout of the current paper based forms are detrimental to a Ray’s business. Using the current Rental Record form as an example, problems can occur when searching for a customer. As there is no unique customer, identification searching for a customer by name could become confusing when two customers have the same name.

4.0 System Requirements

As this is the first computerised system for Ray's business; a stand-alone system would be most efficient. This would keep the hardware and technical support costs to a minimum but would also mean reservations could only be made via a staff member of staff. This system could then be upgraded to a web-based system at a later date; enabling customers' to make their own reservations online, and providing the possibility of developing a mobile application and linking the database to a website. Other requirements may include:

- A system that enables the hiring department staff to easily store and locate customers' contact information, which includes customer name, address, postcode, and telephone number(s). A contact management system (or CMS) will enable a more hassle-free, productive way of interacting with such information and perhaps is ideal when creating a general enquiries log.

  - The staff of the hiring department can respond to customer enquiries via contact management software. Since many customers use the internet these days and may have more than one email-address, a system like a CMS must have the facility or flexibility to store multiple email-addresses for every individual contact.
  - Other records that Ray's business will need to store are: supplier, dealer, employee, maintenance and bike. All of these records will need a unique ID. For example a staff number for payroll or a bike ID to keep track of the rental history for that bike. Each record must be editable so that a record can be deleted, added or changed.

- The ability to maintain customer transaction details of bike rentals including card or cash payment, date of rental, date of return and any faults that have occurred.
- A system that is capable of alerting or providing a warning to Ray and his technicians that the inventory of spare parts is running low or high. It's the sort of operational-level information required as service or repair work will be carried out on a regular basis. This would negate any likely situation of over or under ordering specific bike parts.
- A system whereby enables the departments to collaborate together and perform cross platform activities, for example, the hiring department sends information about bike faults reported by customers to the maintenance department. This is important since staff both respective departments will handling customer issues on a regular basis.
- It may be ideal to have an EPOS (electronic point of sale) system in place as they can perform calculations of total sales, generate receipts and vouchers to customers as well as providing details of transactions to suppliers if necessary. It's a fast and efficient method of dealing goods and services with customers.
- The system should include a standard set of queries relating to all of the records stored on the database. For example, you should be able to search for a customer by the unique customer ID, search for current reservations or for a particular bike part currently in stock.

5.0 Management Reports

Whiteley (2013:145) described management information as:

"An information output from an Information System to aid staff in effective decision-making in the management of the organization"

Management information reports are very important for all business and can be useful in many ways. These could include:

- Easy access to information about the business when it is required.
- It helps to protect valuable information about the business and its customers.
- Vivid view of accountability records of the business.
- It gives room for development as it points out areas that need improvements.
- It uses pictures/images for clearer explanations (graphical representations).

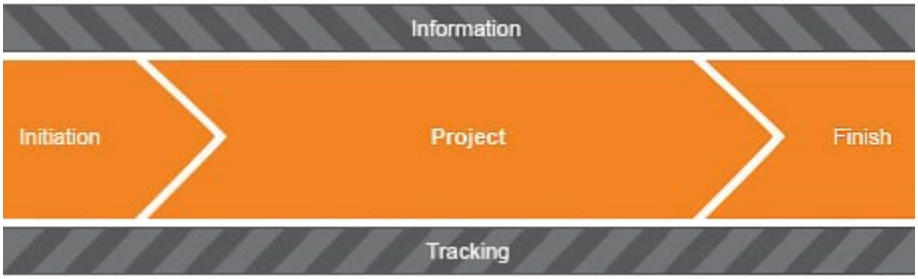


Fig 1.0

Types of Management Information Report:

- Analysis Report
- Exception Report
- Ad hoc Report
- Key Target Report

5.1 Analysis Reports

An analysis report is a type of management information report, which enables the user to analyse data by comparison. The analysed data can be used by all levels of management and can be utilized by higher management to predict trends or to analyse the profits of the company. This is a very basic management report, which is typically in a grid or table. Analysis reports usually focus on overall totals, which can sometimes be misleading to management. For example, a report on sales figures could show a store as being down on their profits compared to the previous month. Theoretically, this could be due to an unforeseen closure of the shop or a limited supply of stock, but this valid reason for reduced profits would not show on the analysis report. Therefore, misleading management into thinking this reduction of profits is due to terrible shop management. An example of this has been shown in figure 1.1.

Sales analysis by region 03 Apr 14 (IR01)

	January	February	March
South-east England	157,000	169,000	172,000
South-west England	97,000	103,000	94,000
Central England	232,000	263,000	187,000
North England	67,000	87,000	91,000
Scotland	63,000	72,000	84,000
Wales	23,000	24,000	23,000
Total	639,000	718,000	651,000

Figure 1.1 D Whitley (2013) Chapter 7, Pg 149

Whiteley (2013) Analysis Report

Alternatively, this analysis report (figure 1.1) can also be displayed in a graphical format (figure 1.2), which will allow the user to analyse the information faster. However, this type of format can also limit the amount of information available to the user, as more space is required for the formatting.

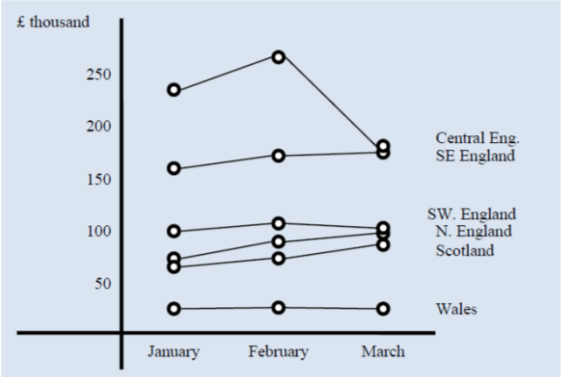


Figure 1.2 D Whitley (2013) Chapter 7, Pg 154

Whiteley (2013) Analysis Report

5.2 Exception Report

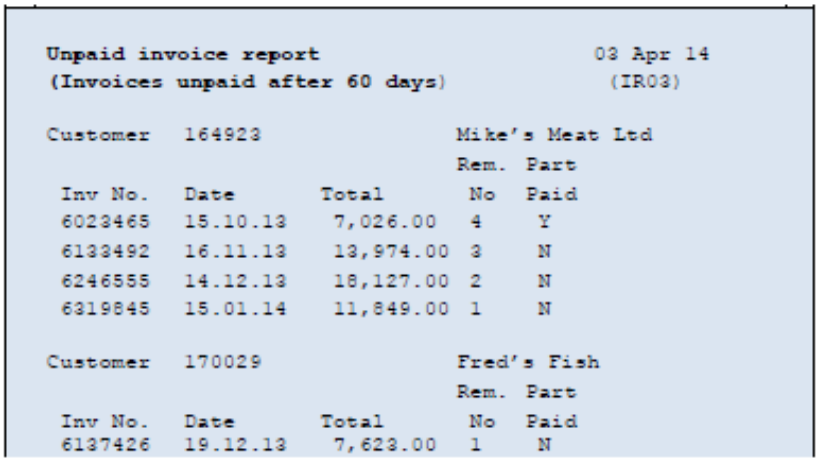
In a nutshell exception reports highlight cases that generally require attention. They notify managers' about unusual or abnormal activities within an organisation. For instance, an exception report for an organisations' HR department may alert management of frequent levels of sick days for certain employees. Similarly, a manager of an auto repair centre might want to reorder certain parts after learning that the inventory was running low. It would present as an area of concern for the manager, so demands immediate action.

Typically performed at the tactical and operational levels of hierarchy, exception reports help managers isolate the problems with other existing matters. The purpose of these is to reduce the effects of information overload, discount any old, unwanted or repetitive information so that managers can respond to problems more effectively and efficiently. In practice, new information or information out of the normal range will be indicated to managers.

However, in any case, there is the likelihood that more information is required for the management to deal with the issues and this enables them to understand why a problem might have occurred. The information in demand is also known as diagnosis information and this will allow managers to perform the diagnosis.

The sample exceptions report below (figure 1.3) illustrates the invoices unpaid after 60 days of two food suppliers:

From the report (figure 1.3) it is clear that Mike's Meat ltd is not meeting the expectations of the supposed retailer (consistently not paying invoices) compared to Fred's report.



Unpaid invoice report 03 Apr 14
(Invoices unpaid after 60 days) (IR03)

Customer 164923 Mike's Meat Ltd

Inv No.	Date	Total	Rem. No	Part Paid
6023465	15.10.13	7,026.00	4	Y
6133492	16.11.13	13,974.00	3	N
6246555	14.12.13	18,127.00	2	N
6319845	15.01.14	11,849.00	1	N

Customer 170029 Fred's Fish

Inv No.	Date	Total	Rem. No	Part Paid
6137426	19.12.13	7,623.00	1	N

Figure 1.3 Exception report – Invoices not paid after certain period

Whiteley (2013) *Exception Report*

Customer Exception Report Date: 13/09/04 Time: 10:57:01 AM

Selection Criteria

Country: FR
Account VAT type: FRSA
Last Invoiced Date:
Analysis A: -
Analysis B: -
Analysis C: -
Analysis D: -
Analysis E: -
Analysis F: -

Cust No.	Cust Name	Cust Alpha	VAT type VAT type Desc.	VAT Registration No.
BIKESHOP	BIKE DISTRIBUTORS LTD.	BIKESHOP		
C001	Hemmingway Engineering LTD + PLC	HEMMING		1234567890
C002	Maybury Motors Ltd	MAYBURY		0987654321
C003	Eastern Motors	EASTERN		
C004	Ashcroft Motors Ltd.	ASHCROFT		
C005	Portland Pumps US Inc.	PORTLAND	Z Zero Rated VAT Type	
C006	Kemper GmbH	KEMPER		
C007	Newton (UK) Ltd	NEWTON		
C008	H D Hugo	HUGO		
C009	Duval S.A.N.V.	DUVAL	Z Zero Rated VAT Type	1234567890
C010	General Motors (UK) Ltd.	GENERAL		

Figure 1.4- A sample customer exception report from SAGE

5.3 Key Target Report

"As someone working on ways to improve organisational performance measures, I know how important it is to look for guidance and the best of what others have done. Those looking to improve their choice and use of key performance indicators will find thought provoking ideas and valuable examples of good practice."

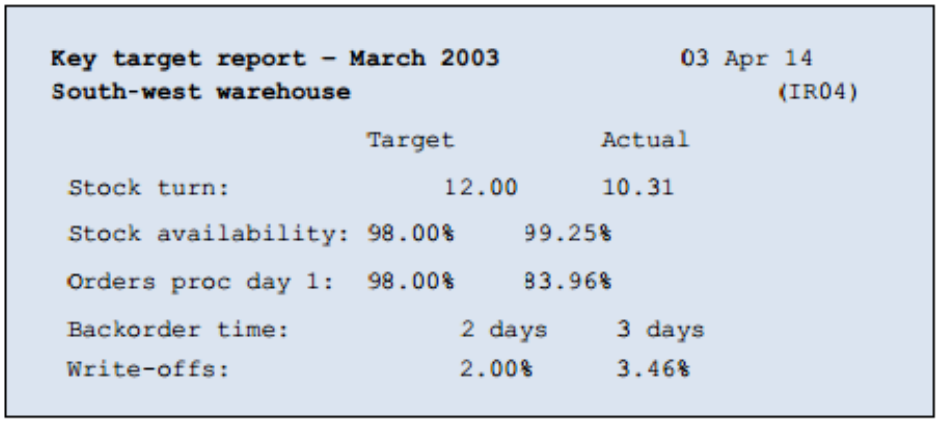
Professor Sir Andrew Likierman

London Business School

As the name implies a Key Target Report, is setting achievable targets, gauges or goals for project by stating expected results and comparing it with the actual results. It is rated or measured in percentage.

'Key target report is a style of management information report where performance data is calculated and set out against per-defined target.'

(Whiteley, 2013)



Key target report – March 2003 03 Apr 14
South-west warehouse (IR04)

	Target	Actual
Stock turn:	12.00	10.31
Stock availability:	98.00%	99.25%
Orders proc day 1:	98.00%	83.96%
Backorder time:	2 days	3 days
Write-offs:	2.00%	3.46%

Fig 1.5 Whiteley (2013) Key Target Report

It has also known as the Key Performance Indicator (KPI) and points out the vital part of the project.

5.4 Ad hoc Reports

An ad-hoc report is a type of management report created on the fly as and when it is required. Ad hoc reports are constructed to meet a specific, unplanned information requirement. They are produced when the information required is not acquirable from regular formal reports.

Ad-hoc reports are time consuming to produce, test and run and may not produce entirely accurate results; by using a fourth-generation query language such as SQL and having the report cross check with a regular report it is possible to greatly reduce the preparation time and increase the accuracy of the report.

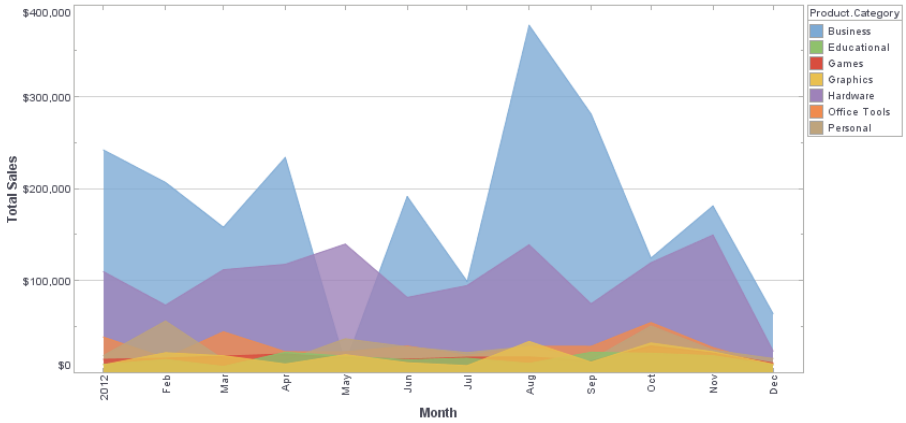
The outcome of ad hoc analysis is typically a statistical model, analytic report, or other type of data summary.

Despite generally only being intended to be run once, ad-hoc reports are often reused and deployed regularly.

In figure 1.6, the information in the table details the overall sales in each different category by each member of staff, whilst the graph displays the sales-by-category over the course of a year.

Sales by Category Report

Feb 14, 2012 5:35:26 PM



	Business	Educational	Games	Graphics	Hardware	Office Tools	Personal	Grand Total
Annie	$591,625	$64,656	$56,165	$49,825	$496,930	$120,146	$100,752	**$1,480,099**
Eric	$481,750	$43,375	$48,397	$68,764	$375,190	$90,436	$88,108	**$1,196,020**
Robert	$582,750	$35,391	$39,841	$46,183	$228,370	$72,780	$69,083	**$1,074,398**
Sue	$510,250	$31,995	$57,196	$35,515	$131,120	$51,327	$59,390	**$876,793**

Fig 1.6 Sales by Category Ad-Hoc Report

6.0 Management reports that will be of use to Ray's Rental

6.1 Monthly Profit and Loss Analysis Report

Ray's business would benefit from a profit and loss analysis report, as it will enable Ray to analyse the cost involved in running his business and compare them against the profits of his business. Ray will then be able to calculate the problem areas of his business and take steps to prevent the mounting costs.

	Months							
	January 13	February 13	March 13	April 13	May 13	June 13		TOTAL
	Income							
Rentals	£ 10,000.00	£ 5,500.00	£ 16,000.00	£ 25,500.00	£ 40,000.00	£ 60,000.00		£ 157,000.00
Sale of bikes	£ 1,000.00	£ 500.00	£ 700.00	£ 200.00	£ 100.00	£ 120.00		£ 2,620.00
	Outgoings							
Wages	£ 3,000.00	£ 3,000.00	£ 3,000.00	£ 3,000.00	£ 3,000.00	£ 3,000.00		£ 18,000.00
Maintenance & Office Costs	£ 1,250.00	£ 1,000.00	£ 2,000.00	£ 2,200.00	£ 2,500.00	£ 3,000.00		£ 11,950.00
Totals	£ 5,750.00	£ 2,000.00	£ 11,700.00	£ 20,500.00	£ 34,900.00	£ 54,120.00		£ 129,670

Figure 2.1 is an example of the Monthly Profit and Loss Analysis Report may look like

The Monthly Profit and Loss Analysis Report would be more useful as an interactive report. When selecting a section, a new report will become available providing more information for that section. Using the Rentals section as an example, this section could be broken down to provide the monthly rental figures for each type of bike.

	Months							
	January	February	March	April	May	June		TOTAL
Bike Type								
Mountain Bike	£ 4,000.00	£ 1,500.00	£ 5,000.00	£ 10,500.00	£ 15,000.00	£ 20,000.00		£ 56,000.00
Road Bike	£ 2,000.00	£ 3,000.00	£ 1,000.00	£ 5,000.00	£ 5,000.00	£ 10,000.00		£ 26,000.00
Tandem	£ 4,000.00	£ 1,000.00	£ 10,000.00	£ 10,000.00	£ 20,000.00	£ 30,000.00		£ 75,000.00
Total	£10,000.00	£5,500.00	£16,000.00	£25,500.00	£40,000.00	£60,000.00		£ 157,000.00

Figure 2.2 is an example of the Rentals section of the interactive Analysis Report.

Other examples could include the monthly income per type of bike or the amount of faults per bike type.

6.2 Maintenance History Exception Report

The following table below is a sample exceptional report of Ray's Rental maintenance history. It will alert Ray of any possible delays or servicing that is pending. Work must be completed to their desired target and if the target is not met then immediate action will be taken.

Maintenance History					
Ref No.	Fault type	Fault date	Action	Action date(within 7 days)	Action taken Yes or No
R001	puncture - rear	15/02/2012	New inner tube	20/02/2012	Y
R002	Chain	23/02/2012	New chain	25/02/2012	Y
R003	Split saddle	04/03/2012	New Saddle	PENDING	N
R004	brakes	18/03/2012	New Brakes	PENDING	N
R005	N/A	12/04/2012	Normal service	12/04/2012	Y

Fig 2.3 Ray's Rentals Maintenance History Exception Report

Unique code – not only assigned to every bike, but also be used as the customer's unique ID/reference Number. Ensures that Ray and his staff are not confused when storing or locating information on a database.

Highlighted in red to present as a warning to Ray – data is outside of normal range (beyond the 7-day limit), hence necessary action will be taken.

6.3 Bike spare parts exception report

The exception report below is a model example of what Ray and his technicians might use to monitor the stock of certain bike components. It eradicates some of the problems that Ray was suffering from (.i.e. over ordering, parts delivered or not etc.) .Furthermore it highlights whether a particular component is in-stock or out-of-stock and whether an order has been placed. More importantly, a specified quantity of the order has been set, immediately suggesting to Ray that parts are unlikely to be over-ordered. Considering the fact that there is "no real check" of order deliveries, it is vital that Ray and his technicians use this information to avoid excessive re-ordering.

Product inventory					
Part type	**REF.NO.**	**Stock/Out Of Stock**	**Order Placed**	**Order Quantity**	**Order Delivered**
Tyre tube	RR12551	IN-STOCK	N	N/A	N/A
brakes	RR12552	IN-STOCK	Y	12	DEL.
saddle	RR12553	IN-STOCK	N	N/A	N/A
chain	RR12554	OUT-STOCK	Y	5	DEL.
Gear cables	RR12555	OUT-STOCK	Y	10	DEL.
Crank bolt	RR12556	IN-STOCK	Y	20	DEL.
Rim Tape	RR12557	OUT-STOCK	Y	14	DEL.

Fig 2.4 Ray's Rentals Bike spare parts Exception Report

Data item highlighted red suggests inventory is running low – (i.e. Limited stock or out of stock). Hence, Ray has placed an order. Although orders placed are subject to availability of the parts, Ray can still place orders for items that are in stock.

6.4 Quarterly Key Target Report

With the help of key target report, Ray's rentals would be able to set targets for its business and figure out the areas that need improvement.

Key target report – (July to September 2013)						
Ray's Rentals						
				Reviewed date	10/10/2013	
	Target			**Actual**		
	large & standard male	large & standard Female	Child	large & standard male	large & standard Female	Child
Bikes Availability	92%	90%	95%	87%	91%	98%
Bikes sold	10%	7%	3%	6%	11%	0%
Decommissioned bikes	2%	4%	1%	4%	1%	3%

Fig 2.5 Ray's Rentals quarterly Target Report

Bikes Availability – This is the total number of bikes that are available to the customer at that particular time.

Bikes sold - This is the proportion of the bikes in stock that have been sold in the quarter.

Decommissioned bikes - This is the percentage of the bikes that were undergoing maintenance and were not available to rent out to customers.

This report is based on the sizes of bikes that was hired /rented out to customers and not on the classes (mountain, road and tandem.

6.5 Ad hoc Reports

The following graph is an ad-hoc report showing the rental popularity of bike classifications over a period of 12 months. This report will help Ray's Rental to identify which bikes classes they need to stock most during specific periods of the year.

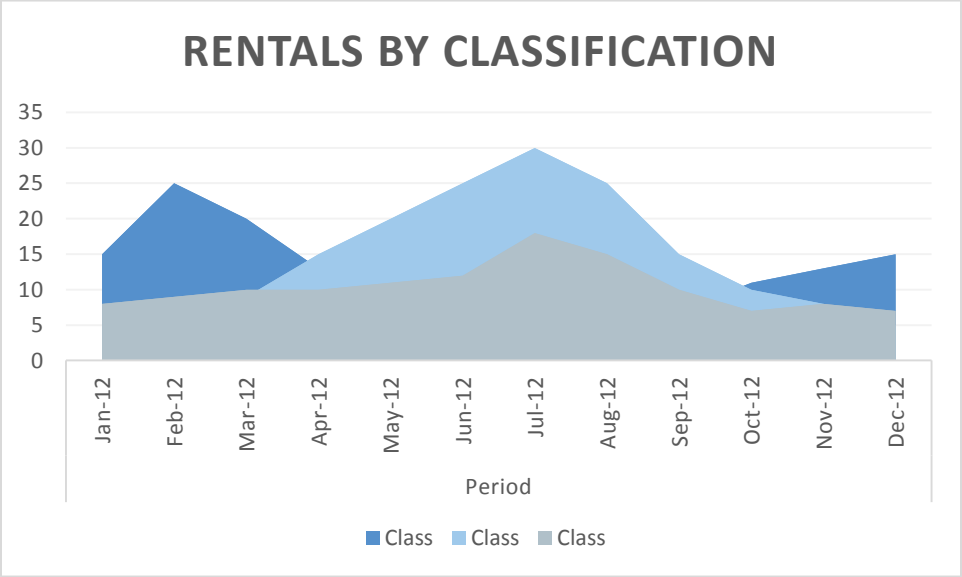


Fig 2.6 Rentals by Classification Report

The report clearly shows that mountain bikes are most popular during the winter months, whilst Road bikes are most popular during the summer months. It also shows that BMX bikes are reasonably popular throughout the year, peaking during the summer-holiday period.

7.0 Part One Conclusion

According to the findings, there are a series of problems with the existing paper-based systems. Clearly, a computerised database system will help eliminate the issues that Ray's Rentals currently suffers from. The management reports researched will ensure Ray's business operates efficiently and effectively, providing they take advantage of the reports.

All management reports will provide beneficial factors for the business, however using a combination of these reports would help to sustain the longevity of the business.

Information Systems Project

Part Two

Systems Analysis & Design

8.0 Part Two Introduction

In section 9.0, a Use Case diagram of the desired information system for Ray's Rentals has been designed. This is a group Use Case diagram; the diagrams of individual team members have been combined to create a more substantial diagram. Commentary has been added to provide further description of the thought process involved.

Section 10.0 provides a comprehensive Use Case specification for most use cases outlined in the Use Case diagram. It has also been outlined which team member has completed each specification.

In section 11.0 the rental and bike form for Ray's rentals have been normalised and the RDA of each form has been listed. For each form an ERD has been created which have then been combined.

In section 12.0 an overall team ERD has been delineated. The ERD is a combination of all ERDs which have been created in previous sections; additional attributes have been added where the team thought it was necessary.

A short conclusion of the analysis and design process is in section 13.0.

9.0 Use Case Diagram

In figure 3.0 a group use case diagram has been illustrated. This is the final diagram; many changes have been discussed in this iterative design process.

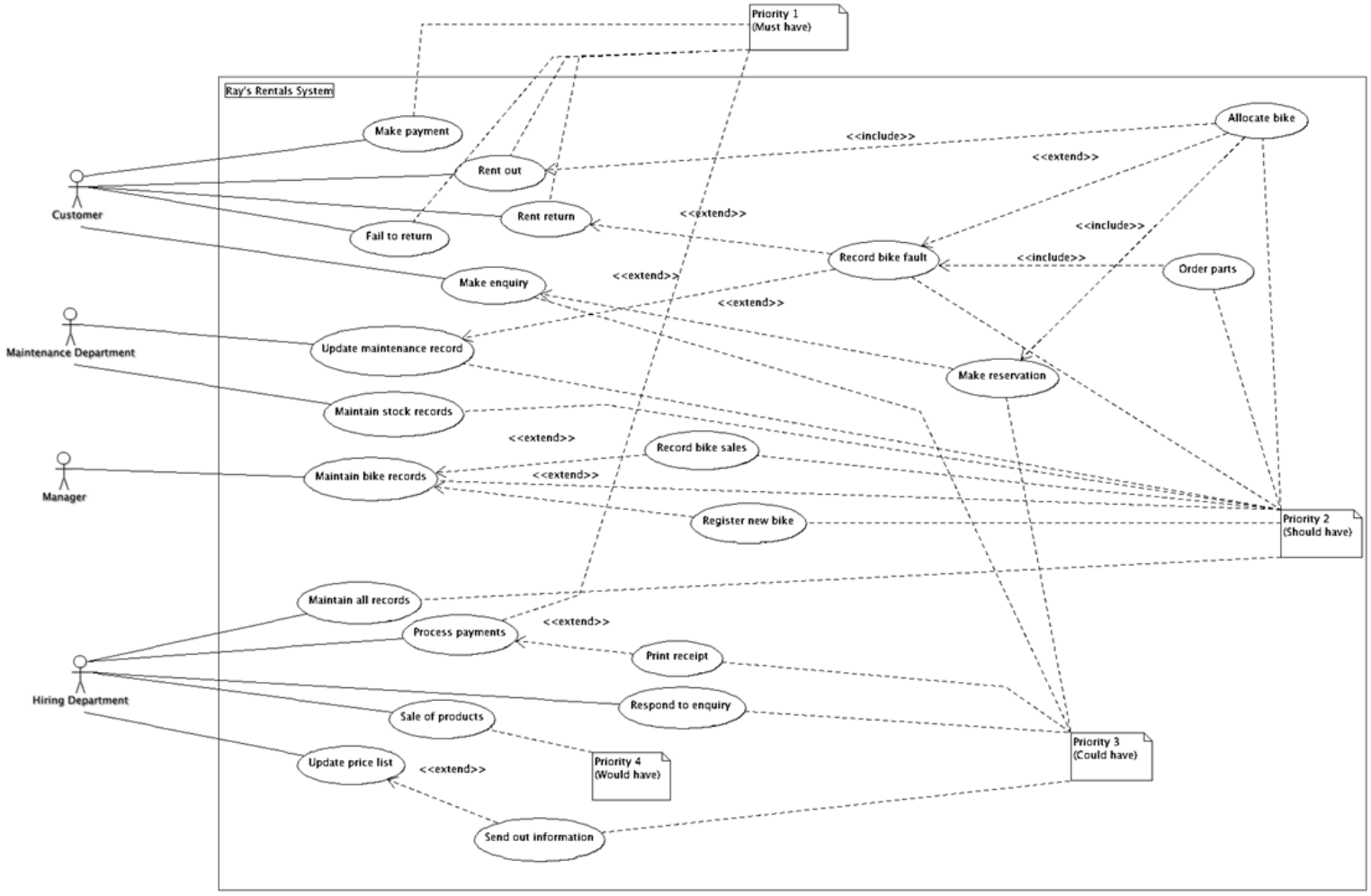


Figure 3.0 use case specification

9.1 Commentary

When combining all of the use case diagrams it became clear that the four main actors were: Customer, Maintenance Department, Manager and Hiring Department. These actors were chosen because of their substantial interactions with the system without any duplication from further actors. All of the team members agreed that the maintenance department and hiring department were essential to systems functionality. Amanda and Vamsi outlined the importance of the customer's role. Vamsi also stressed the significance of the Mangers interactions with the system and outlined the uses cases of consequence. Precious suggested a Parts manager as an additional actor. The use cases for the Parts manager were then integrated into the maintenance department actor.

Vamsi, Amanda and Precious all made contributions to the use cases and dependencies. Amanda then outlined the prioritisation for each use case and illustrated the reviewed use case diagram. Vamsi completed all the use case specifications for Customer. Amanda was responsible for all the use case specifications for the Hire department. Precious completed all the use case specifications for Manager and Maintenance Department.

In the process of designing the use case diagram, it was identified that prioritisation of the systems functionality is vital for non-technical users. This enables them to understand the purpose of the system and to contribute in the design process. This is iterative process, which allows for easy design modifications. For example adding an extension to clarify a use case or removing a duplicated use case.

10.0 Use Case Specification

In this section use cases have been explained using use case specifications. The name of the individual completing each use case is clearly marked. Included in each use case specification is an ERD of the entities needed for that use case.

The ERD helps to give a better understanding of which data is necessary for the task being completed.

An activity diagram has been added to make the alternative and primary paths easier to understand.

Use Case: Maintain All Records

Owner: Hiring Department

Name: Amanda McIntosh
Version: 1.2
Date: 26/11/13

Pre-Conditions

There must be a previous record or the information needed to create a new record.
The computer must be turned on.
There must be a network connection.
All hardware must be working, for example there must be a working keyboard and mouse.

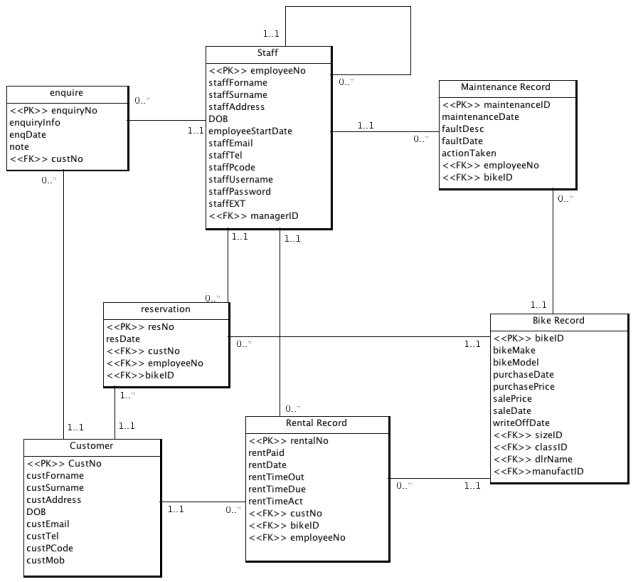
Post-Conditions

The record will be added to the system and will then be available to other members of staff. If a record has been changed the previous record will no longer be available on the system. The record will also show the last member of staff who edited the record and the date the changes were made.

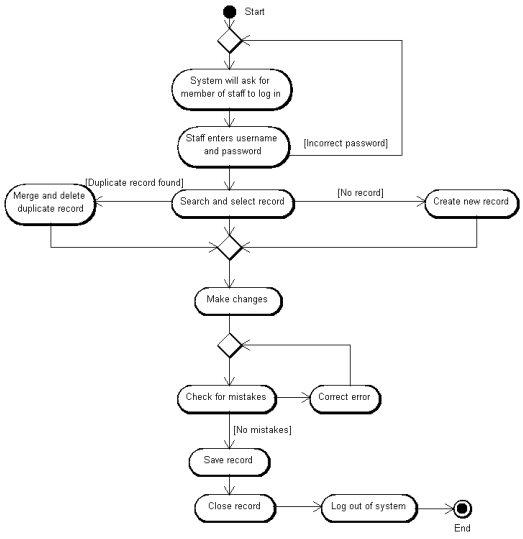
Primary Path

1) **The system will prompt for a username and password.**
2) **A member of staff must enter their username and password to gain access to the system.**
3) **The member of staff must access the record needed by searching and selecting the record.**
4) **The member of staff will make the necessary changes or updates using the information they have been provided with.**
5) **The member of staff will then check the record for mistakes and change where necessary.**

6) **The member of staff will then save the record.**
7) **The record will then be closed.**
8) **The staff member will log out of the system.**



Alternate Path



1.1 **No previous record**

If there is no record available then the employee will have to create a new record in step three.

1.2 Record duplication

If there is a record duplication then the member of staff must merge the records and delete the duplicate, this will happen in step three.

1.3 log in failure

In step two; if the member of staff entered an incorrect username or password then the system will prompt them to re-enter their username and password, this loop will continue until a correct username and password have been entered.

Notes

This can apply to all records that the maintenance team must update. For example, this process could apply to updating a customer record or a rental record.

Use Case: Process Payments

Owner: Hiring Department

Name: Amanda McIntosh

Version: 1.2

Date: 26/11/13

Pre-Conditions

The customer must have selected the bike they want to hire and the bike must be available.

The customer has provided the hire staff with all their details as well as an imprint of their card and has been provide with a customer number.

The customer must specify the length of time they want to rent the bike for and will have been provided with an estimated price.

The customer must have rented the bike for a certain length of time and is now returning the bike to the store.

Post-Conditions

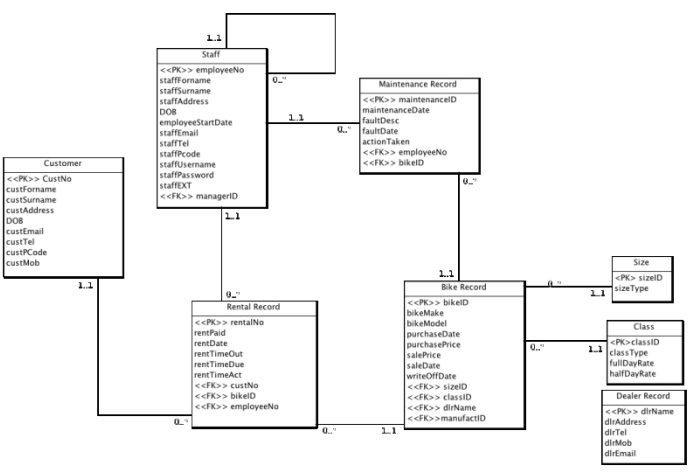
The customer will leave the store with a receipt as proof of payment.

The bike will be returned to the storeroom and made available for renal once the bike has been marked as returned.

The record will state which member of staff edited the record last.

Primary Path

1) **A member of the hire department must log onto the system with their username and password.**
2) **The customer must provide the hiring department with their customer number. The hire department will enter the customer number and access the rental record for the customer.**
3) **The member of the hire department will then update the record to bike returned.**
4) **The system will then calculate the hire price based on the classification of bike and the length of time the bike has been used for.**
5) **The customer will then be asked to report any faults.**
6) **The customer will pay calculated amount.**
7) **The member of the hire team will then update the record to 'paid'.**
8) **The hire department will then print a receipt for the customer.**
9) **The record will then be saved and closed.**



Alternate Path

1.1 Payment failure

If the payment fails then the cashier will ask the customer to pay by an alternative method in part six. If this is not possible then an invoice will be sent to the customer and the process will end.

1.2 Rental time overdue

The original estimated time will be amended in step four to compensate for the extra time.

1.3 Customer complaint

If there is a valid complaint and a manager has approved a 20% discount will be added to the bill in step four.

1.4 Failure to return bike

In the first instance the customer will be contacted and asked to return the bike; a late fee will be added to part four. If the customer cannot be contacted then their details will be reported to the police and the bike will be reported stolen, this information will then be added to the records in part 4.

1.5 log in failure

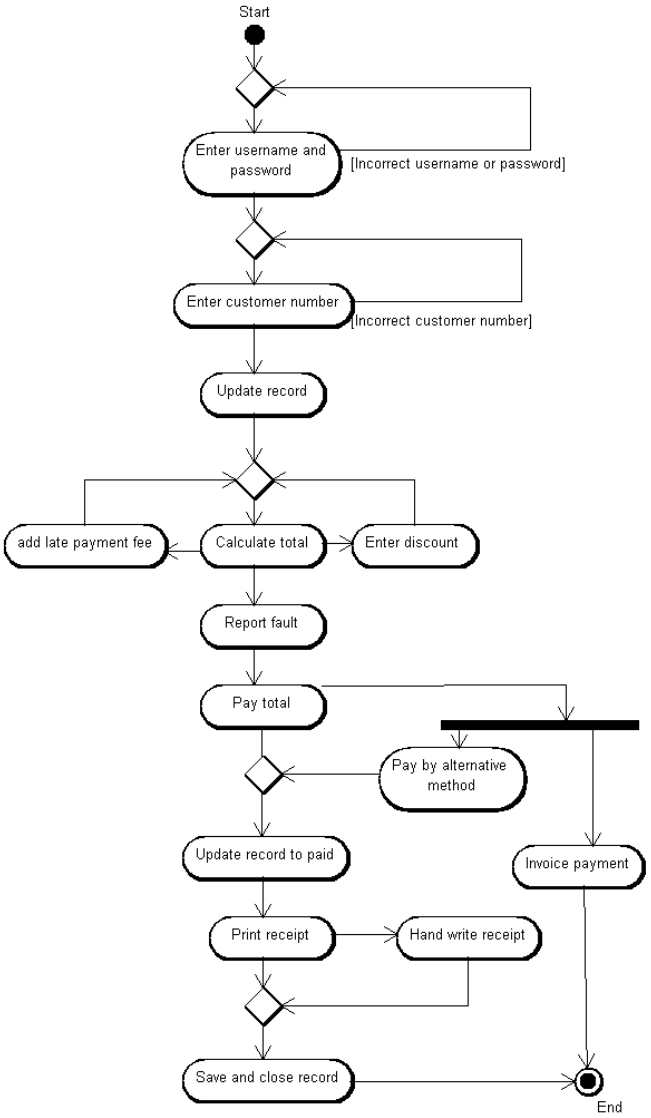
If the member of staff entered the incorrect username or password in step one, then the system will prompt them to re-enter their username and password.

1.6 Incorrect customer number

If the member of staff enters an incorrect customer number the system will prompt the user to re-enter the customer number in step two.

1.7 Receipt print failure

The member of staff will need to hand write a receipt for the customer in step eight.



Notes

Use Case: Respond to Enquiry

Owner: Hiring Department

Name: Amanda McIntosh

Version: 1.2

Date: 26/11/13

Pre-Conditions

A customer must have enquired by post, telephone, email or in store.

A member of staff must be available to respond to the enquiry.

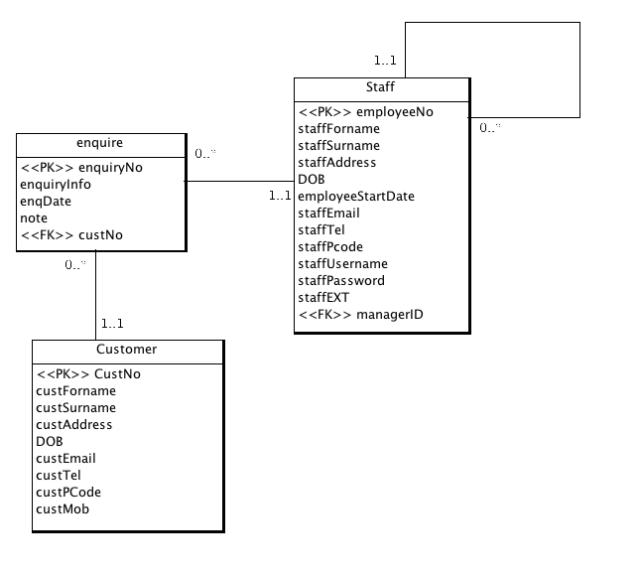
There must be a network connection and an available working PC.

Post-Conditions

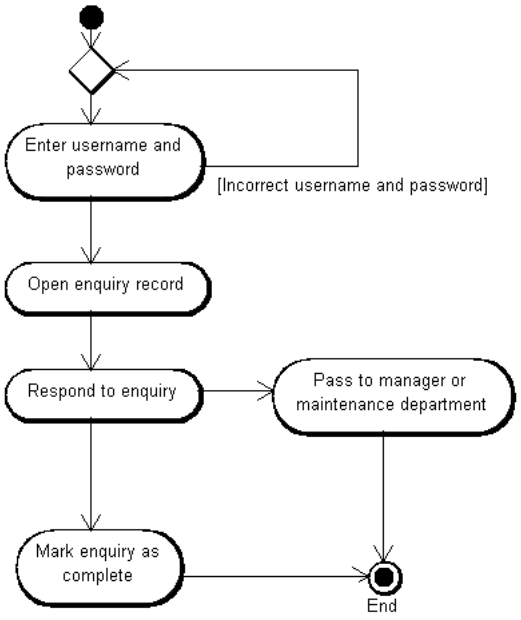
The customer will have received an answer to their question via the method of communication they chose.

Primary Path

1) **The member of staff must log into the system using their username and password.**
2) **The member of staff must access the enquiry records.**
3) **The member of staff will select the enquiry to deal with based on the date received.**
4) **The hire department will respond to the customer by phone, email or post.**
5) **The hire department will then mark the enquiry as completed.**



Alternate Path



1.1 Response unavailable

If the hiring department cannot handle the enquiry they may need to refer it to a manager or the maintenance team.

1.2 Customer is in store

The hire department will provide the customer with the information needed instantaneously.

Notes

Use Case: Sale of Products

Owner: Hiring Department

Name: Amanda McIntosh

Version: 1.2

Date: 26/11/13

Pre-Conditions

The product must be in stock.

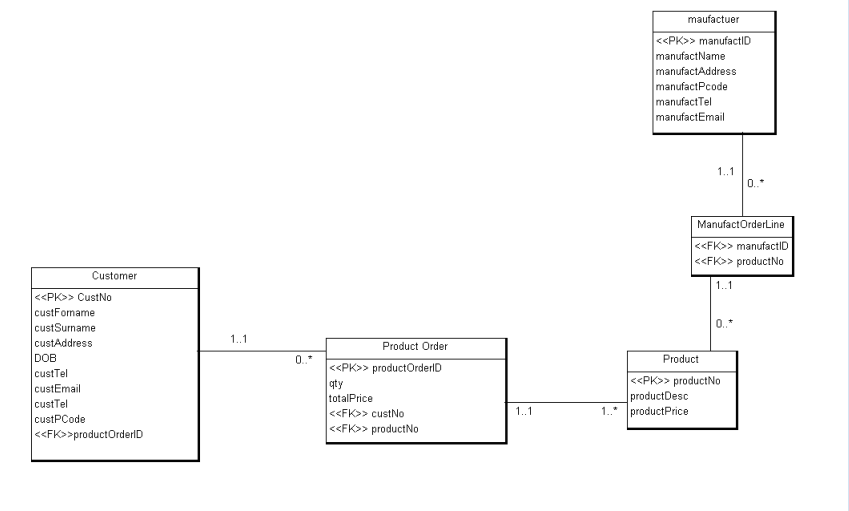
The customer must browse the store and select a product.

Post-Conditions

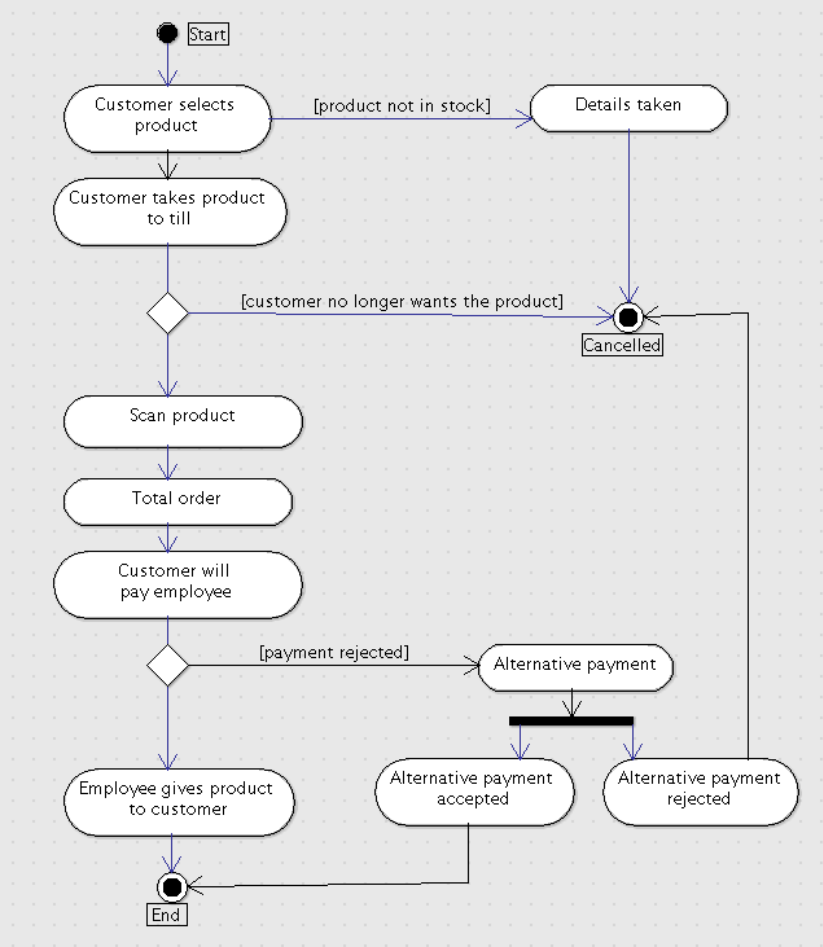
The customer will leave the store with the product they wanted and a receipt for their purchase.

Primary Path

1) **The customer must take the product to the till**
2) **The cashier will log on to the system.**
3) **A new product order record will be automatically be created by the system.**
4) **The cashier will scan the product and state the price of the product to the customer.**
5) **The system will then total the order**
6) **The cashier will ask for a method of payment.**
7) **The customer will then pay the cashier by cash or card.**
8) **The cashier will then print a receipt and give it to the customer**
9) **The order record will then be completed and closed.**
10) **The cashier will then ask the customer if they want a bag and place the product in a bag.**



Alternate Path



1.1 Product not wanted

If the customer no longer wants the product; the cashier will cancel the order and return the product to its place in the shop. If the payment has already been processed then the cashier will then need to refund the payment first.

1.2 Payment failure

If the payment does not process the cashier will ask the customer for an alternative payment, if there is not alternative payment available to the customer then the cashier will cancel the order and return the product to the shelf.

1.3 More items

If the customer wants to purchase more than one item then the cashier will scan each product before completing the order and taking a payment.

1.4 Product not in stock

If the item the customer wants is not in stock then they will need to ask the cashier to order the product. The customer will then need to provide their contact details and the store will contact the customer when the product is in stock. This information will be stores as a customer enquiry.

Notes

This is a desired use case in the event of Ray expanding his business to sell products.

Use Case: Update price list

Owner: Hiring Department

Name: Amanda McIntosh
Version: 1.0
Date: 26/11/13

Pre-Conditions

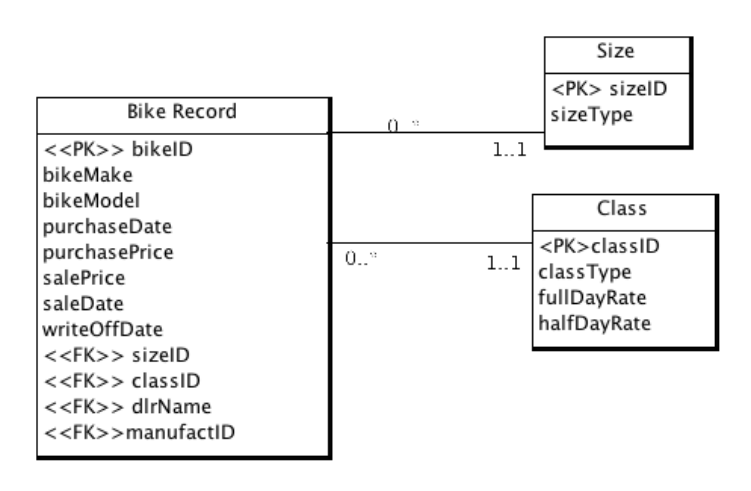
The member of staff must have the access rights to access the record.
There must be a valid network connection.

Post-Conditions

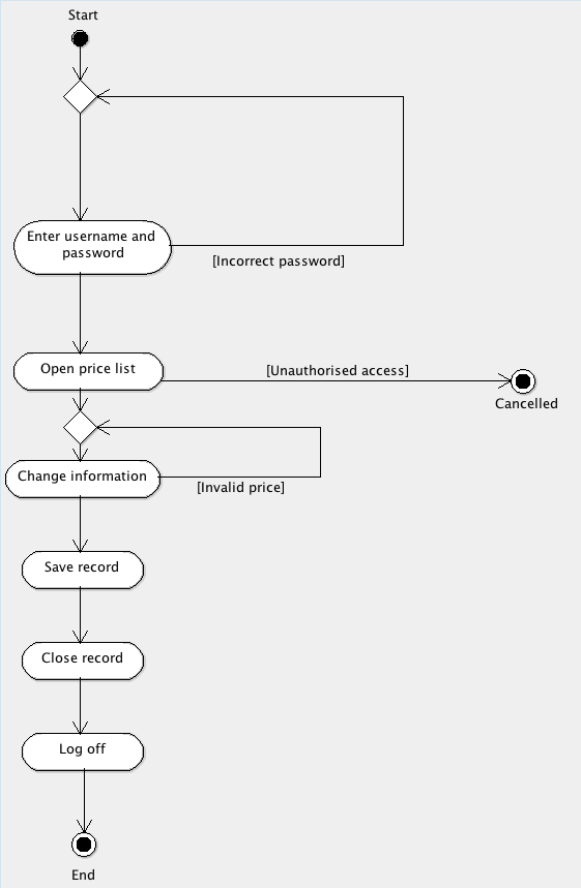
The price list will be changed on the system and will be available for other users to view and print.

Primary Path

1) **Enter username and password to log onto the system.**
2) **Open the class record to edit the rental rates which are saved.**
3) **Change information.**
4) **Save record.**
5) **Close record.**
6) **Log off the system.**



Alternate Path



1.1 **Unauthorised to update**

If the member of staff does not have the permission right to update the record then the system will advise the user to change their setting through their systems administrator. The process will be cancelled.

1.2 **Invalid username or password**

The system will prompt the user to re-enter their username or password.

1.3 **Invalid price**

To prevent mistakes on this record their will be a validation check, if the user enters an invalid figure, for example letters are included, then the system will prompt the user to revise their modifications. This will be part of step 4.

Notes

Use Case: Send out information

Owner: Hiring Department

Name: Amanda McIntosh
Version: 1.0
Date: 26/11/13

Pre-Conditions

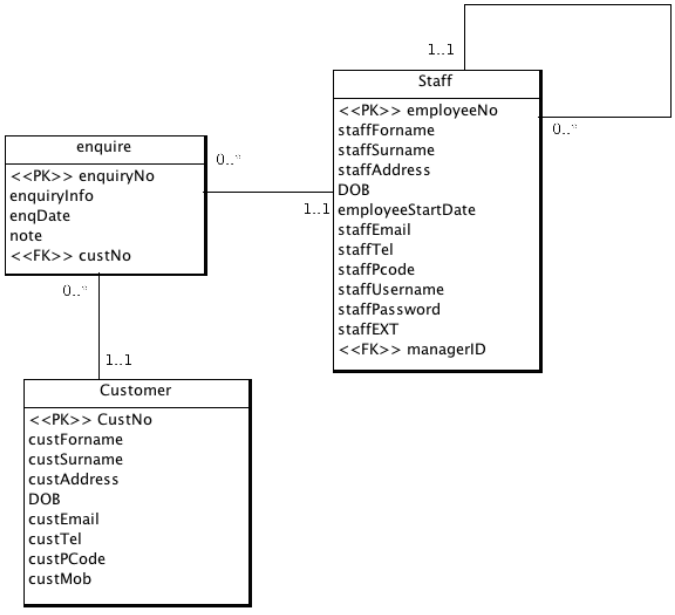
A member of staff must be logged into the system.
A member of staff must have opened an enquiry record and must need to respond by post.

Post-Conditions

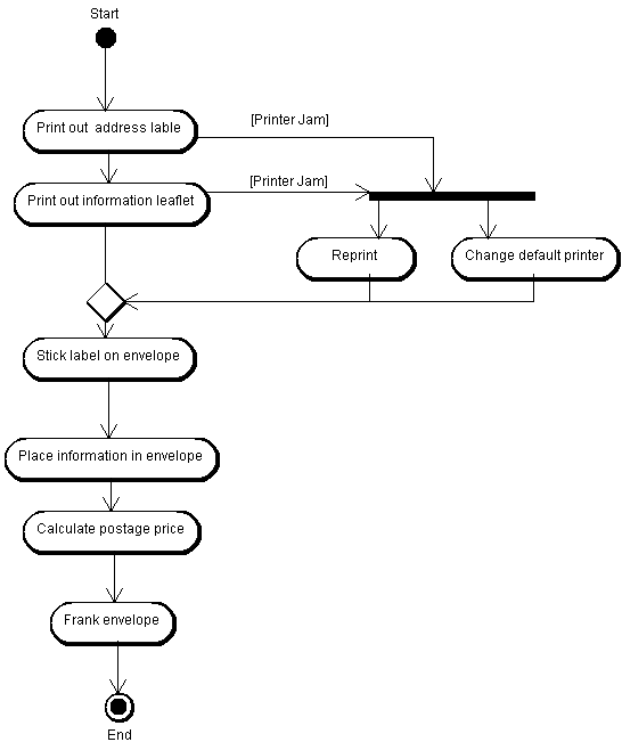
The franked envelope will be taken to the post office and the customer will receive the information in a few days.

Primary Path

1. **Automatically print out address label for envelope using the customers' record.**
2. **Print out information leaflet.**
3. **Stick label on envelope**
4. **Place information in envelope**
5. **Calculate postage price based on the size and weight of the envelope.**
6. **Frank envelope.**



Alternate Path



1.1 Printer jam

In the event of a printer jam a member of staff must fix the paper jam and then re-print the information. If the printer cannot be fixed another printer must be used.

1.2 Incorrect customer information

If the member of staff notices a mistake on the label then they must enter that customer's record and change their information. They can then proceed to printing the address label again.

Notes

Use Case: Rent out

Owner: Customer Name: Vamsi Version:1.0 Date:28/11/13

Pre-Conditions

Customer can rent out bike in person at Ray's Rentals.

Specific bikes in stock are subject to availability.

Post-Conditions

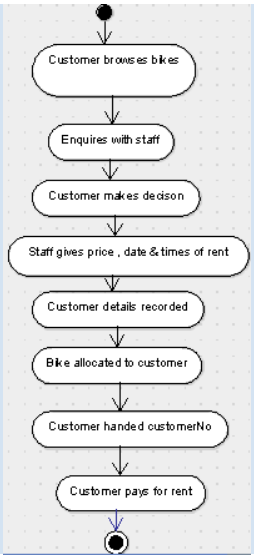
Customer leaves Ray's Rentals with the rent due dates , times and price details . Bike is recorded as "rentTimeOut" in the database(Rental record)

Primary Path

1. **Customer begins choosing a bike of their choice**
2. **Enquires with hiring department staff over bikes**
3. **Customer makes their decision**

Hiring staff gives price for customers chosen date and time

4.
5. **Hiring staff enters customer details into database**
6. **Bike is allocated to customer**
7. **Customer handed a "customerNo" for future verification**
8. **Customer makes payment**



Alternate Path

Customer does not want to pay for rent before taking bike

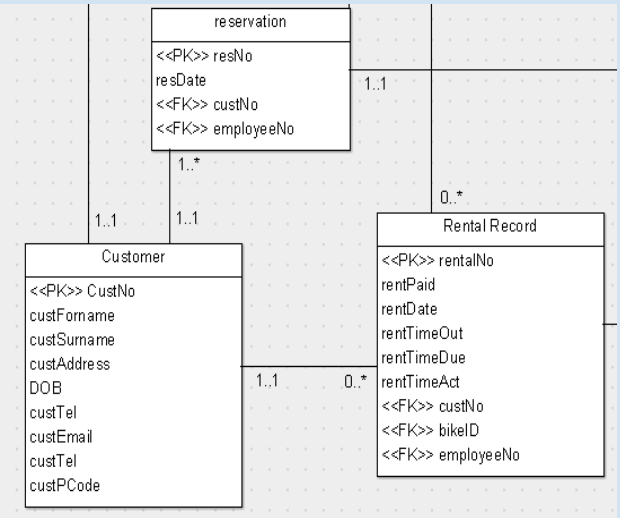
1. **Hiring staff gives option to pay after returning bike**
2. **Customer returns bike**
3. **Proceeds with payment**

Preferred Rent dates/times not available

1. **Hiring staff reserves bike after customer requests reservation**
2. **Customer handed a "reserveNo"**
3. **Hiring staff will inform customer when dates/times are available**

Information Systems Project

Notes



Use Case: Rent Return

Owner: Customer Name: Vamsi Version:1.0 Date:28/11/13

Pre-Conditions

Customer returns bike to the hiring staff at the required date and time.

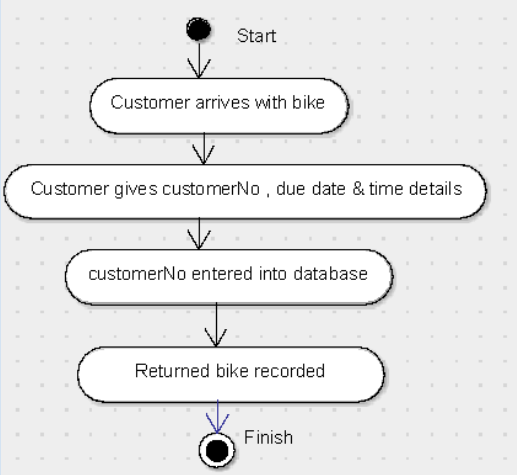
Arrives with "customerNo" to complete bike return

Post-Conditions

Customer leaves Ray's Rental after staff records bike into database with reference to customerNo

Primary Path

1. **Customer arrives with bike**
2. **Provides hiring staff member with customerNo , due date and time details.**
3. **Staff member enters customerNo into database to verify bike due for return**
4. **Bike is taken in and is recorded into database**



Alternate Path

Customer fails to return bike

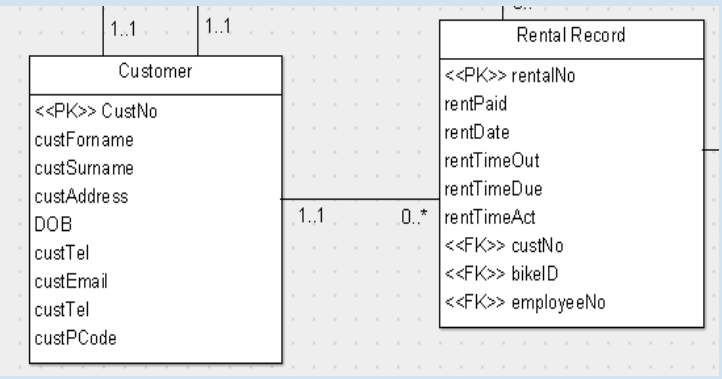
1. **Customer signs a letter of agreement to return bike within 7 working days**
2. **Customer returns bike**
3. **Staff imposes fine/charge on customer if bike not returned within requirements**
4. **Customer accepts fine/charge**

Customer returns bike that has faults

1. **Customer returns bike**

2. **Complains to staff , it has a faulty part**
3. **Staff records bike faults into the database (Maintenance Record)**
4. **Faults reported to maintenance department**

Notes



Use Case: Make enquiry

Owner: Customer **Name: Vamsi** **Version:1.0** **Date:28/11/13**

Pre-Conditions

A variety of enquiry services are made available to customers : by email , phone or in person at the Ray's Rentals.

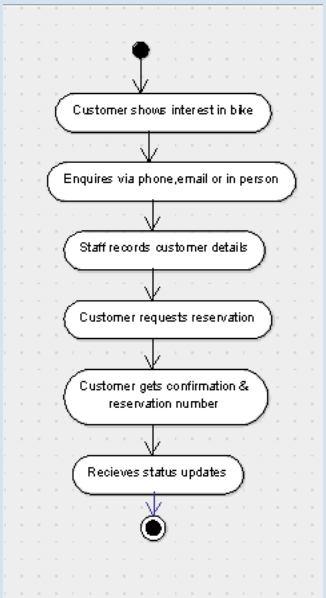
Post-Conditions

Customer receives information regarding the bikes

Hiring staff records customer details into a database after customer expresses interest and will proceed with a reservation.

Primary Path

1. **Customer expresses interest in a particular bike**
2. **Seeks for more information about bikes via phone , email or in person**
3. **Hiring staff records customer details into a database after customer confirms interest in bike rent . Customer details include :**
   - **Customer name**
   - **Address**
   - **Post-code**
   - **Phone number(s)**
   - **Email address**
4. **Checks availability of bike**
5. **Customer requests reservation**
6. **Receives confirmation details, including customer reservation number .**
7. **Staff will provide customer with updates regarding the current status of reservation.**



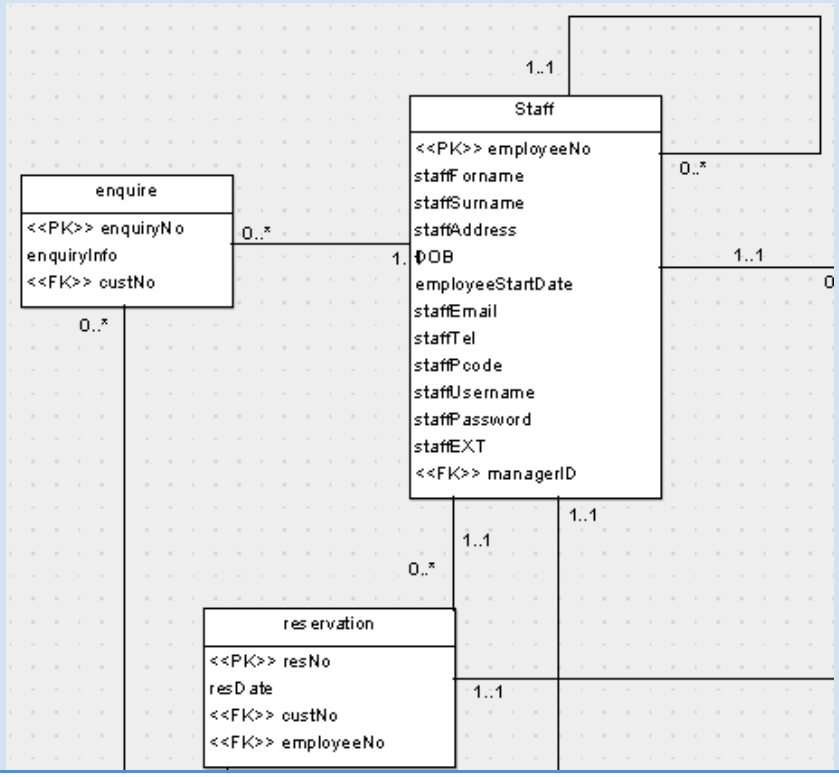
Alternate Path

Email servers fail ; customers unable to enquire or receive confirmation via email

1. **Customer makes phone call**
2. **Customer interested in a particular bike**
3. **Staff records customer details**
4. **Checks availability of bike**
5. **Customer requests reservation**
6. **Receives booking confirmation (inc. reservation No) and updates via SMS**

Information Systems Project

Notes



Use Case: Fail to return

Owner: Customer **Name: Vamsi** **Version:1.0** **Date:28/11/13**

Pre-Conditions

Customer fails to arrive with bike. Customer required to sign a formal letter of agreement by where bike must be returned within 7 working days, otherwise face consequences (fine/charge).

Customer gives customerNo to hiring staff member as requested

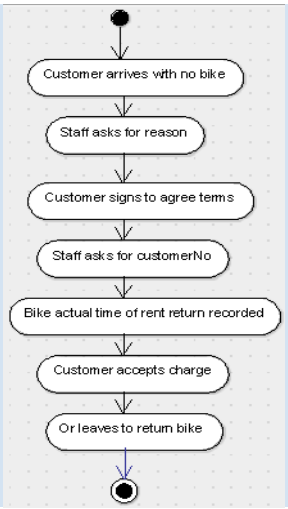
Post-Conditions

Customers signs formal agreement , either accepting charge or leaving Ray's Rental to return bike.

Hiring staff member records bike into database(Rental Record) with reference to customerNo

Primary Path

1. **Customer arrives with no bike**
2. **Hiring staff member asks for an explanation**
3. **Customer required to sign formal letter of agreement**
4. **Staff member asks for customerNo**
5. **Records actual time of bike return into database(Rental Record)**
6. **Customer accepts charge**
7. **Otherwise Customer leaves to return bike within 7 working days**



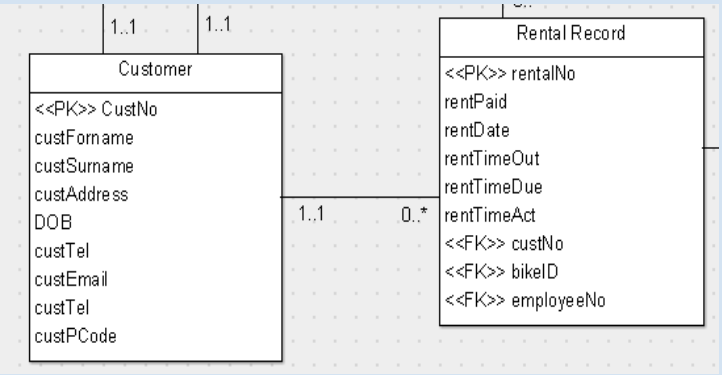
Alternate Path

Customer fails to return bike even after signing agreement

1. **Customer fails to return bike**
2. **Staff member insists customer must accept a greater fine after disobeying rules**
3. **Customer accepts the fine/charge and pays for it**

Information Systems Project

Notes



Use Case: Allocate bike

Owner: Customer **Name: Vamsi** **Version:1.0** **Date:28/11/13**

Pre-Conditions

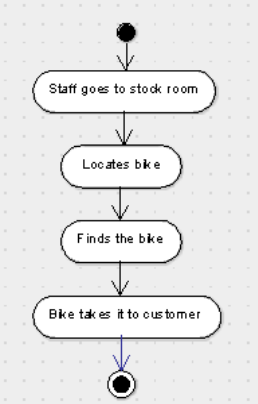
Hiring staff member has the details (i.e. specification – model , make , size) of the customers' preferred bike

Post-Conditions

Customer's bike is available in the stock/store room and is in good condition. Staff member locates bike and takes it to the customer.

Primary Path

1. **Staff member goes to stock room**
2. **Locates bike with reference to details of bike**
3. **Successfully locates bike**
4. **Bike handed to customer for further proceedings**

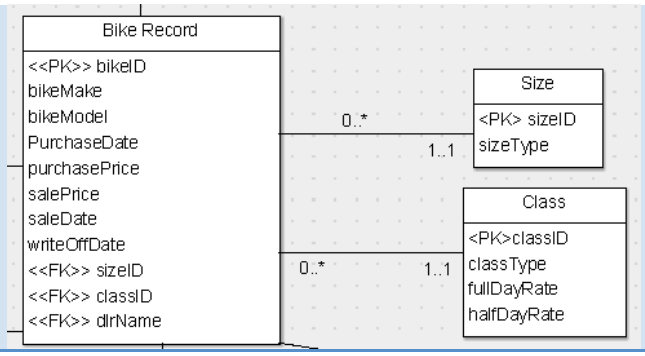


Alternate Path

Customers' chosen bike not available or needs to be serviced/repaired

1. **Staff member looks for an alternative bike in stock**
2. **Otherwise keeps the customers' preferred bike on reserve**
3. **Customer informed when bike is available**

Notes



Use Case :Make Payment

Owner: Customer **Name: Vamsi** **Version:1.0** **Date:28/11/13**

Pre-Conditions

Customer has all the necessary information to make payment for renting bike (i.e. booking confirmation, customerNo , reserveNo etc.) .

Payment options made available for customer ; cash or credit card

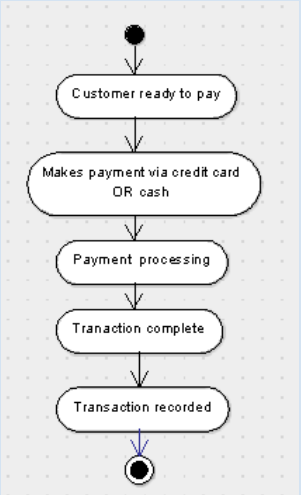
Post-Conditions

Customer makes payment successfully.

Customers' pay for the rent or purchases is recorded on database (Rental Record)

Primary Path

1. **Customer ready to make payment**
2. **Staff member gives option for customer pay either by credit card or cash**
3. **Payment processing**
4. **Transaction complete**
5. **Transaction recorded on database (Rental Record)**



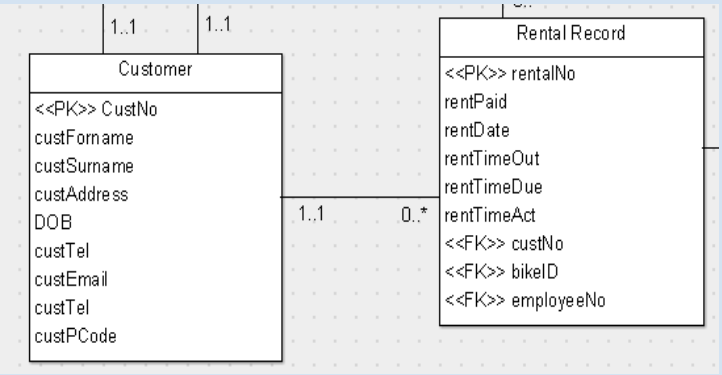
Alternate Path

EPOS system/Chip & pin machine temporarily out of order (not working/malfunctioning)

1. **Payment system not working**
2. **Customer cannot complete payment**
3. **Staff member request payment by cash only**
4. **Customer make payment by cash**

Information Systems Project

Notes



Use Case: Update Maintenance Record

Owner: Maintenance Department

Name: Precious Igbinosun

Version:1.0

Date:07/12/13

Pre-Conditions

Bikes must have a bike record, which contains the maintenance history of the bikes.

The bikes should be due for servicing (this is a regular maintenance that is carried out over a set period of time. This could be monthly, yearly etc.

A fault has been reported or detected on a bike.

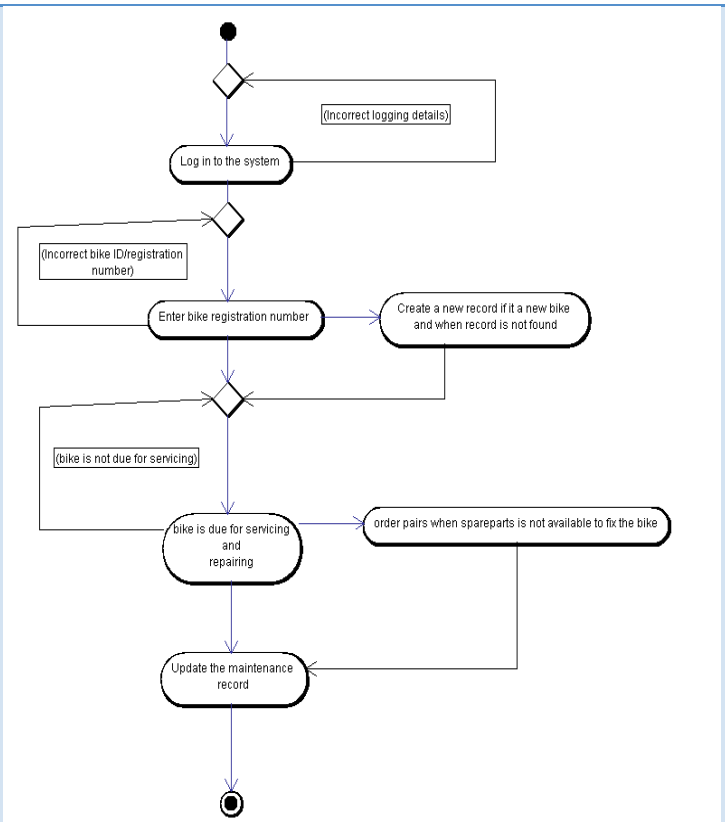
Post-Conditions

The bike record is updated.

The bike will be repaired and can be rented out to customers.

Primary Path

1. **The maintenance staff must enter a staff ID to access the system.**
2. **This use case can only begin either when a bike is due for service or faulty.**
3. **The bike ID is identified and search for the bike record is checked, to know when the bike was lasted serviced or repaired.**
4. **Carry out a standard service this include general strip-down or rebuilding the bike with new tyre.**
5. **Lubricate the moving parts of the bike.**
6. **Order parts for bike if needed e.g. brakes, chain belts, cables etc.**
7. **Report bikes that are due for sales.**
8. **Record all the maintenance carried out on the bike in the bike record for that particular bike.**



Alternate Path

1. **Bike is not due for servicing.**
2. **There is no spare parts in the warehouse**
3. **There is a delay in the delivery of parts.**

Notes

Use Case: Maintain Stock Record

Owner: Maintenance Department
Name: Precious Igbinosun
Version:1.0
Date:07/12/13

Pre-Conditions

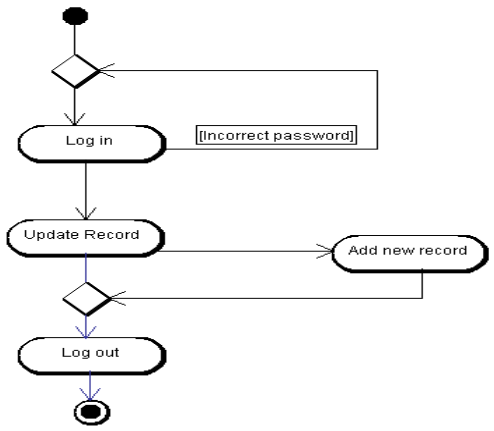
A new bike has been purchased.
Spare parts are out of stock.

Post-Conditions

The stock record will be updated

Primary Path

1. **Member of staff will log into the system, using the staff ID and password.**
2. **Update the stock record.**
3. **Log out of the system.**



Alternate Path

1. **Wrong username and password**
   - **it displays an error message and prompt the user to try again to contact the helpdesk.**
2. **Stock record does not exist**
   - **Create a new record for stock that don't exist**

Use Case: Maintain Bike Record

Owner: Manager

Name: Precious Igbinosun

Version:1.0

Date:07/12/13

Pre-Conditions

Only the manager can maintain a bike record and must have a logging in details, which should be a staff number.

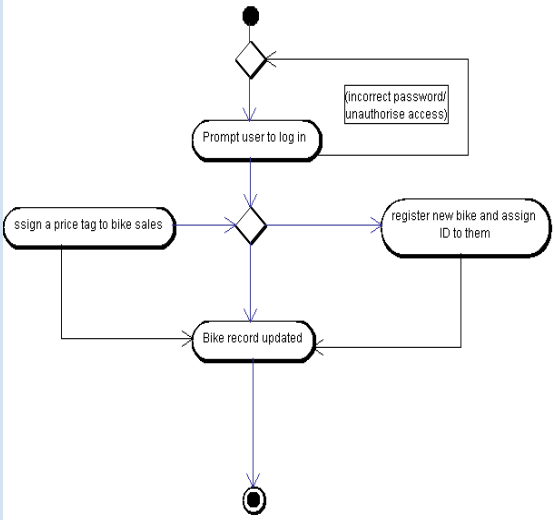
- **When new bikes are purchased**
- **Bike is due for sale**

Post-Conditions

- **A new record will be created**
- **Bike is sold out**
- **Record is updated**

Primary Path

- **The user log in with username**
  Password
- **If it a new bike, user has to register the bike and create a record for it**
- **When a bike is to be sold out, user assign price to the bike and save record**
- **Record is updated**



Alternative path

- **Get authorise access from the management staff**

Notes

This actor (manager) can also access other processes in this system

11.0 RDA

In this section the Bike Record form has been normalised (figure 4.0) to the third normal form. A Bottom Up ERD has been included as further description (figure 4.1). The Rental Record has also been normalised to the third normal form in figure 5.0. A Bottom Up ERD for this form has also been included to determine the data which is needed in the system (figure 5.1).

11.1 Bike Record RDA

Un- Normalised	First Normalisation	Second Normalisation	Third Normalisation
bikeNo	bikeNo	bikeNo	bikeNo
bikeModel	bikeModel	bikeModel	bikeModel
bikeClassification	bikeClassification	bikeClassification	classificationID*
classificationID	classificationID	classificationID	sizeID*
bikeSize	bikeSize	bikeSize	bikePurchaseDate
sizeID	sizeID	sizeID	bikePrice
bikePurchaseDate	bikePurchaseDate	bikePurchaseDate	maufacName*
bikePrice	bikePrice	bikePrice	dispoDate
manufacName	maufacName	maufacName	dlrName*
manufacAddress	manufacAddress	manufacAddress	salePrice
manufacPostcode	manufacPostcode	manufacPostcode	
manufacTel	manufacTel	manufacTel	bikeNo*
dispoDate	dispoDate	dispoDate	maintRef
dlrName	dlrName	dlrName	faultDetails
dlrAddress	dlrAddress	dlrAddress	faultDate
dlrTel	dlrTel	dlrTel	faultAction
salePrice	salePrice	salePrice	faultActionDate
maintRef			
faultDetails	bikeNo*	bikeNo*	classificationID
faultDate	maintRef	maintRef	bikeClassification
faultAction	faultDetails	faultDetails	
faultActionDate	faultDate	faultDate	sizeID
	faultAction	faultAction	bikeSize
	faultActionDate	faultActionDate	
			maufacName
			manufacAddress
			manufacPostcode
			manufacTel
			dlrName
			dlrAddress
			dlrTel

Figure 4.0 RDA for Bike record

NOTE - classificationID and sizeID have been added to the un-normalised section to provide class and size with a unique identifier.

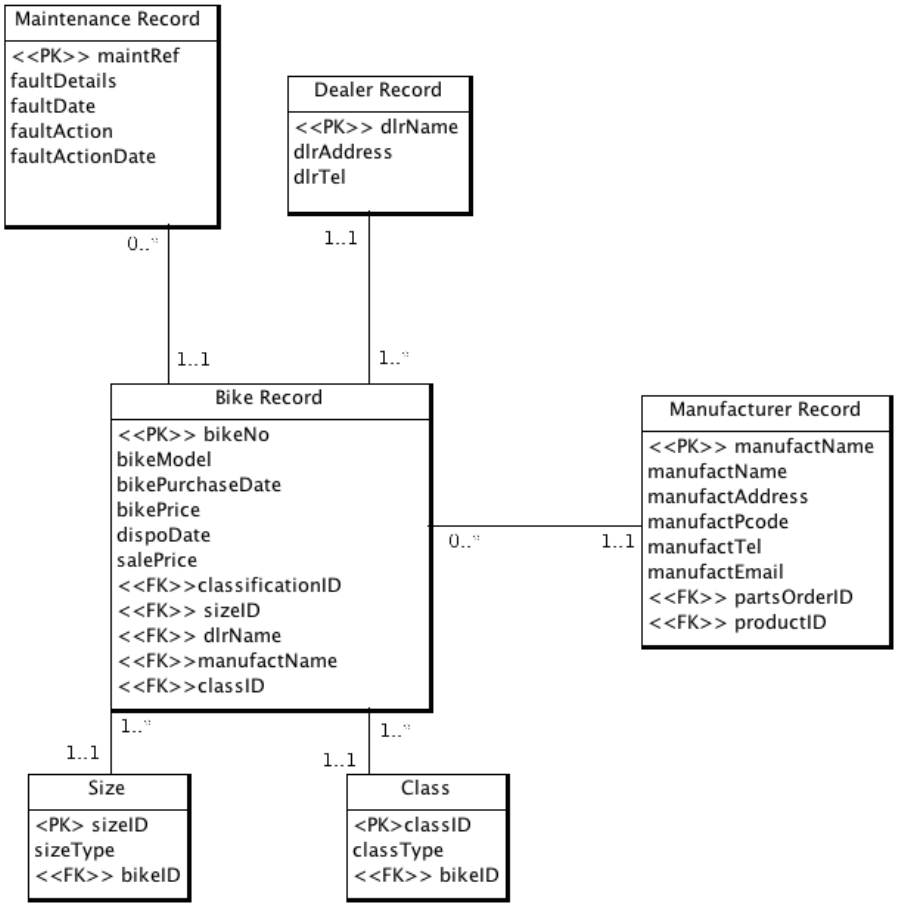


Figure 4.1 Bottom Up ERD for Bike Record RDA

Figure 4.1 is an ERD for the paper based Bike Record currently being used by Ray's Rentals. All of the entities and attributes have been deemed as essential by the paper records already available to Ray's business. After reviewing the normalised data, it became clear that six entities were needed in order to correctly store the date needed for the Bike Records. The manufacturer relationship was added based on the assumption that manufacturer record could be added without purchasing from them. This explained the 0..* relationship as a manufacturer could also have sold the business many bikes.

11.2 Rental Record RDA

Un-Normalised	First Normalisation	Second Normalisation	Third Normalisation
bikeNo bikeName bikeType bikeSize rentNo rentDate rentTimeOut rentTimeDue rentTimeAct custNo custName custAddress custPostcode custTel rentAmountPaid	bikeNo bikeName bikeType bikeSize bikeNo* rentNo rentDate rentTimeOut rentTimeDue rentTimeAct custNo custName custAddress custPostcode custTel rentAmountPaid	bikeNo bikeName bikeType bikeSize bikeNo* rentNo rentDate rentTimeOut rentTimeDue rentTimeAct custNo custName custAddress custPostcode custTel rentAmountPaid	bikeNo bikeName bikeType bikeSize bikeNo* rentNo rentDate rentTimeOut rentTimeDue rentTimeAct custNo* rentAmountPaid custNo custName custAddress custPostcode custTel

Figure 5.0 RDA for Rental Record

NOTE - rentNo and custNo have been added to the un-normalised section to provide the rental record and the customer with a unique identifier.

Figure 5.1 is an ERD for the normalised Rental Record form. The Rental Record only underlines a small quantity of attributes. After normalisation only three entities were required in this ERD. The relationships were added based on the assumption that a customer's record can be saved on the system without them needing to rent a bike and a bike could be added to the system without it being rented, for example a new bike will have no rental records attached.

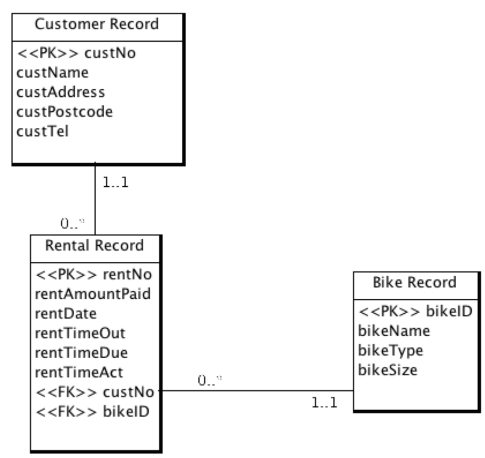


Figure 5.1 Bottom Up ERD for Rental Record RDA

11.3 RDA ERD

Figure 6.0 is the combined ERD for both normalised form from section 11.0. A UML notation has been used.

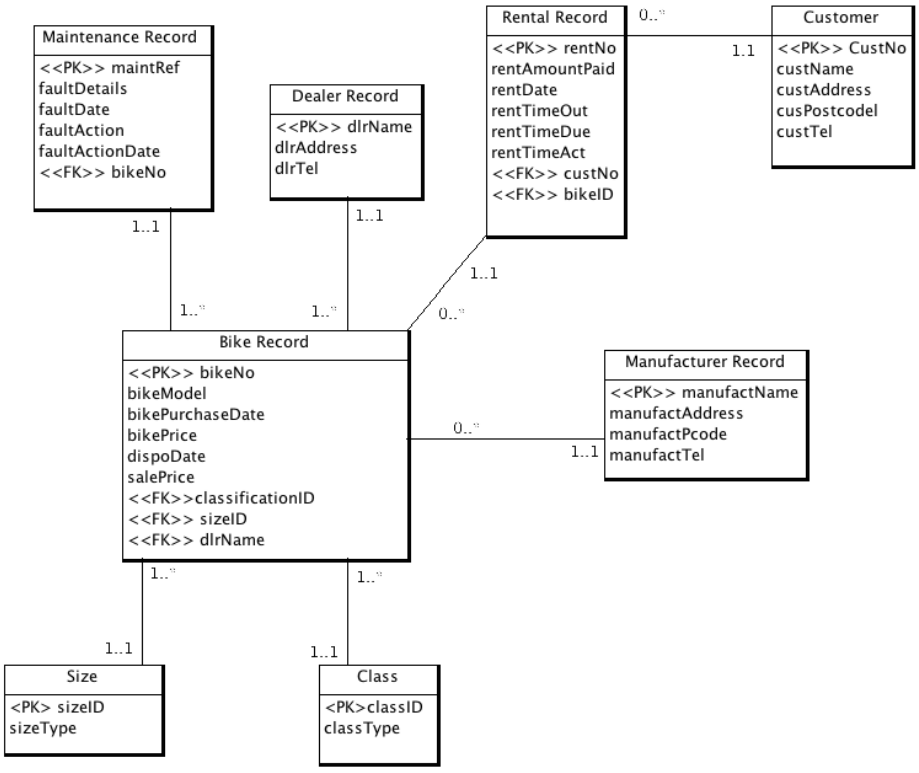


Figure 6.0 Bottom Up ERD of merged RDAs

When combining the two bottom up ERDs all of the bike attributes we merged into one entity. The relationships and other entities remained the same.

12.0 ERD

This section compromises of a group ERD in UML notation for Ray's Rentals.

NOTE – The Primary Keys in manufactID are combined to provide the entity with a unique identification, this means that manufactID and partNo are primary and foreign keys and combined they are a composite key.

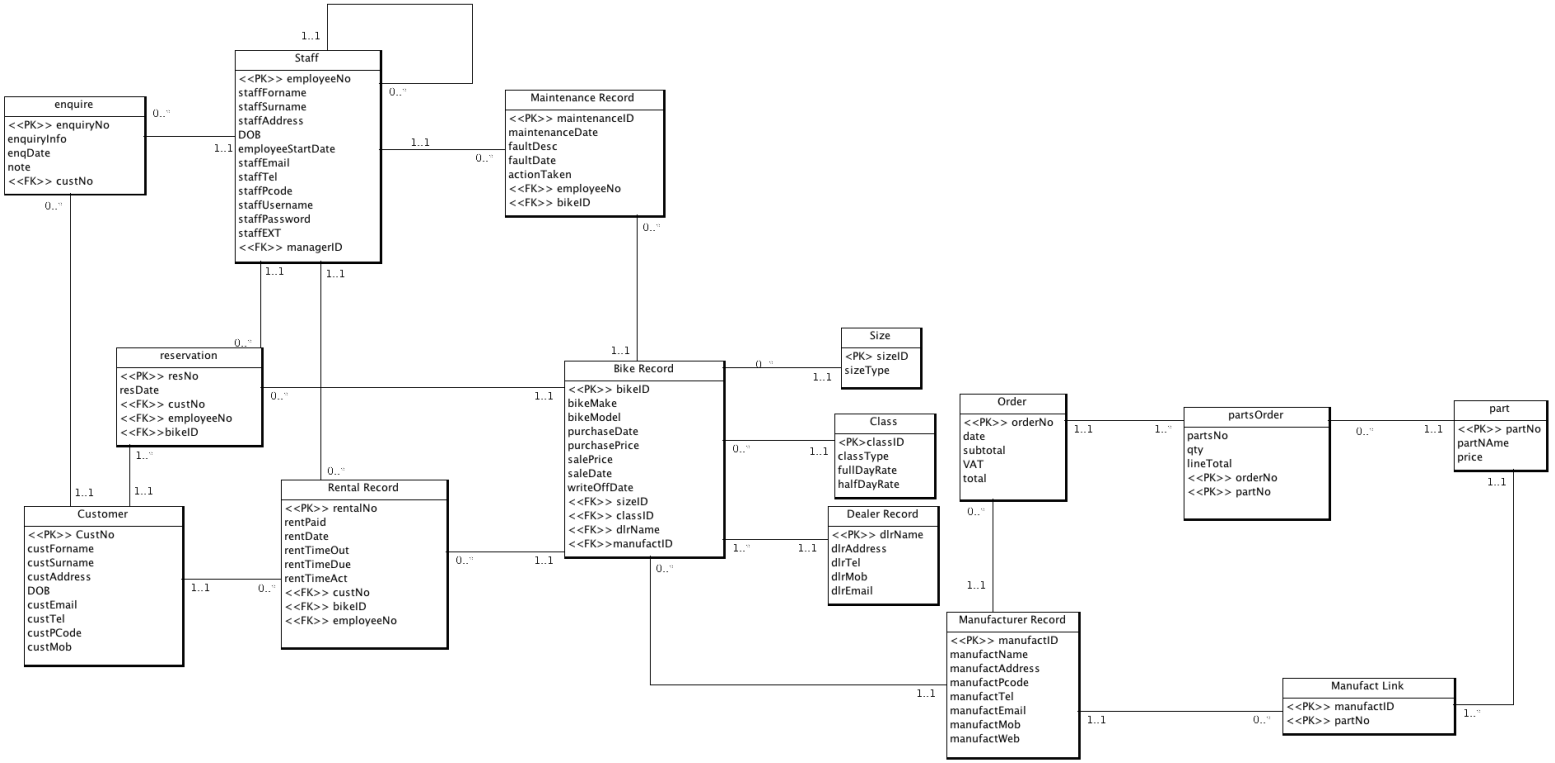


Figure 7.0 Group ERD

12.1 Commentary

When analysing the top down ERDs used for the use case specification it became clear that there were too many unnecessary entities which could result in the system becoming confusing. On further inspection of the combined bottom up ERDs (figure 4.1 & 5.1) it was suggested that this should be the base for our final ERD as this was a simpler diagram which could easily be developed.

Adding the staff entity enabled the system to store all the staff information needed and also added an extra level of security as it enables the storage and use of usernames and passwords for each member of staff. The reservation entity was added as it was clear from the case study that reservations of bikes were essential to Ray's business. The manufacturer records were extended to include a parts entity as this would enable faster re-ordering of parts for a specific bike. This emerged as a many to many relationship and was resolved by adding a link between manufacturer and part record. An order entity was also attached to the manufacture record, which resulted in a many to many relationship, this was resolved by creating a partOrder link.

The entity for buying merchandise were not added as it was not needed for the working system at this point in time.

Any zero to many relationships work on the assumption that the record can be saved independently.

This was a group effort and all team members took an active role.

13.0 Part Two Conclusion

The use case diagram provides a means of communication between the non-technical users and the system development team, whilst giving a clear overview of the main processes which must be included within the system.

The Entity Relationship Diagrams provide a graphical representation of the essential data which must be stored by the system. The relationships identified make the flow of data clearer to the user. For every entity a Primary Keys has been identified to make the data stored unique and useful. Foreign keys have also been outlined to created consistency of the tuples.

These diagrams make the system easy to understand, this will help non-technical users to easily provide relevant information to the system design team. This will ultimately result in a superior system at the end of the system development lifecycle. This design process is a crucial method of highlighting the data needed by the system.

Information Systems Project

Part Three

DB Design and Oracle Implementation

14.0 Part Three Introduction

Section 15.0 consists of an updated ERD and a small commentary explaining the changes and their relevance to the SQL script.

In section 16.0 a data dictionary has been created for entity by analysing the final ERD. The main purpose of this task was to create a standardised document to refer to when creating the SQL code. The data type and length has been specified for each attribute and any constraints have been outlined.

Section 17.0 provides documentation of the running SQL queries, illustrated by screenshots and a small description of each query has been included.

Each member of the team has provided a short statement in section 18.0, describing what they have learnt in database design and Oracle implementation.

15.0 Updated ERD

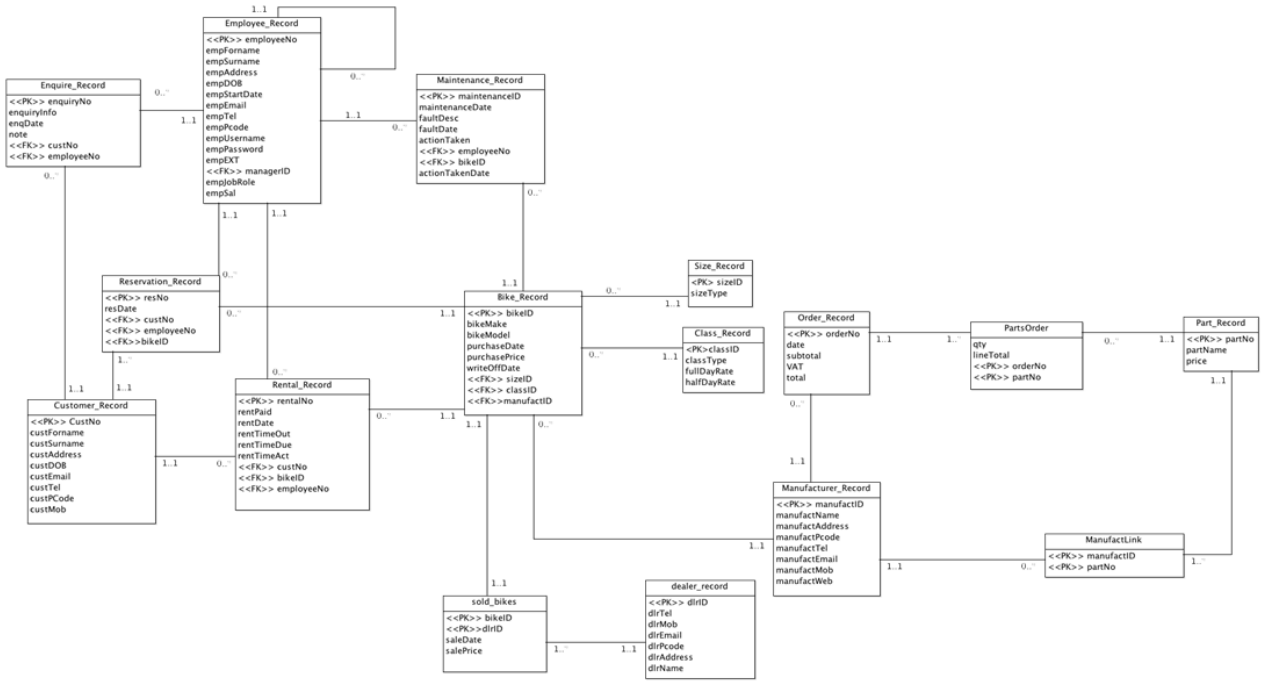


Figure 8.0 Final ERD

15.1 Commentary

Minor alterations have been made to the final ERD (Figure 8.0); unnecessary capitalised letters and foreign keys have been removed. Staff has been changed to employee or 'emp' in each attribute to decrease the length of the attribute name and to add consistency which will be advantageous when writing the SQL statements. The name of most entities has been changed to include a postfix of _Record, this will prevent errors when running the SQL code as some previously named entities are SQL Syntax. Salary was added to staff to aid in the calculation of monthly salary and to enhance the database's querying functionality.

Some extra attributes were added to provide extra detail to each record, a staff job role was provided to establish the role of each member of staff; action taken date, enquiry date and enquiry notes were added to provide further detail.

16.0 Date Dictionary

Data Dictionary: enquire_record

Attribute Name	Key Type	FK Table	FK Column	Data Type	Length	Constraint
enquiryNo	PK			NUMBER	10	PRIMARY KEY
enquiryInfo				VARCHAR2	100	NOT NULL
enquiryDate				DATE	dd-mm-yyyy	
note				VARCHAR2	100	
custNo	FK	customer_record	custNo	NUMBER	10	FOREIGN KEY

Data Dictionary: customer_record

Attribute Name	Key Type	FK Table	FK Column	Data Type	Length	Constraint
custNo	PK			NUMBER	10	PRIMARY KEY
custForname				VARCHAR2	20	
custSurname				VARCHAR2	20	NOT NULL
custAddress				VARCHAR2	50	NOT NULL
custPCode				VARCHAR2	8	NOT NULL
custDOB				DATE	dd-mm-yyyy	
custEmail				VARCHAR2	30	UNIQUE, NOT NULL
custTel				VARCHAR2	11	
custMob				VARCHAR2	11	

Data Dictionary: reservation_record

Attribute Name	Key Type	FK Table	FK Column	Data Type	Length	Constraint
resNo	PK			NUMBER	10	PRIMARY KEY
resDate				DATE	dd-mm-yyyy	NOT NULL
custNo	FK	customer_record	custNo	NUMBER	10	FOREIGN KEY
employeeNo	FK	employee_record	employeeNo	NUMBER	10	FOREIGN KEY
bikeID	FK	bike_record	bikeID	NUMBER	10	FOREIGN KEY

Data Dictionary: employee_record

Attribute Name	Key Type	FK Table	FK Column	Data Type	Length	Constraint
employeeNo	PK			NUMBER	10	PRIMARY KEY
empForname				VARCHAR2	20	
empSurname				VARCHAR2	20	NOT NULL
empAddress				VARCHAR2	50	NOT NULL
empPCode				VARCHAR2	8	NOT NULL
empDOB				DATE	dd-mm-yyyy	
empStartDate				DATE	dd-mm-yyyy	
empJobRole				VARCHAR2	20	
empSal				NUMBER	(8,2)	CHECK (empSal < 100000)
empEmail				VARCHAR2	30	UNIQUE
empTel				VARCHAR2	11	
empUsername				VARCHAR2	15	UNIQUE
empPassword				VARCHAR2	15	
empEXT				VARCHAR2	5	
managerID	FK	employee_record	employeeNo	NUMBER	10	FOREIGN KEY

Data Dictionary: rental_record

Attribute Name	Key Type	FK Table	FK Column	Data Type	Length	Constraint
rentalNo	PK			NUMBER	10	PRIMARY KEY
rentPaid				NUMBER(7,2)	7	CHECK >0
rentDate				DATE	dd-mmm-yy	NOT NULL
rentTimeOut				VARCHAR2	5	NOT NULL
rentTimeDue				VARCHAR2	5	
rentTimeAct				VARCHAR2	5	
custNo	FK	customer_record	custNo	NUMBER	10	FOREIGN KEY
bikeID	FK	bike_record	bikeID	NUMBER	10	FOREIGN KEY
employeeNo	FK	employee_record	employeeNo	NUMBER	10	FOREIGN KEY

Data Dictionary: bike_record

Attribute Name	Key Type	FK Table	FK Column	Data Type	Length	Constraint
bikeID	PK			NUMBER	10	PRIMARY KEY
bikeMake				VARCHAR2	15	NOT NULL
bikeModel				VARCHAR2	15	NOT NULL
purchaseDate				DATE	dd-mm-yyyy	NOT NULL
purchasePrice				NUMBER	(7,2)	CHECK >0
writeOffDate				DATE	dd-mm-yyyy	
sizeID	FK	size_record	sizeID	NUMBER	10	FOREIGN KEY
classID	FK	class_record	classID	NUMBER	10	FOREIGN KEY
manufactID	FK	manufacturer_record	manufactID	NUMBER	10	FOREIGN KEY

Data Dictionary: maintenance_record

Attribute Name	Key Type	FK Table	FK Column	Data Type	Length	Constraint
maintenanceID	PK			NUMBER	10	PRIMARY KEY
maintenanceDate				DATE	dd-mm-yyyy	NOT NULL
faultDesc				VARCHAR2	50	
faultDate				DATE	dd-mm-yyyy	NOT NULL
actionTakenDate				DATE	dd-mm-yyyy	
actionTaken				VARCHAR2	2	NOT NULL
employeeNo				NUMBER	10	NOT NULL
bikeID	FK	Bike_Record	bikeID	NUMBER	10	FOREIGN KEY

Data Dictionary: size_record

Attribute Name	Key Type	FK Table	FK Column	Data Type	Length	Constraint
sizeID	PK			NUMBER	10	PRIMARY KEY
sizeType				VARCHAR2	50	NOT NULL

Data Dictionary: dealer_record

Attribute Name	Key Type	FK Table	FK Column	Data Type	Length	Constraint
dlrID	PK			NUMBER	10	PRIMARY KEY
dlrName				VARCHAR2	20	NOT NULL
dlrAddress				VARCHAR2	50	NOT NULL
dlrPCode				VARCHAR2	8	NOT NULL
dlrTel				VARCHAR2	11	
dlrMob				VARCHAR2	11	
dlrEmail				VARCHAR2	30	UNIQUE, NOT NULL

Data Dictionary: manufacturer_record

Attribute Name	Key Type	FK Table	FK Column	Data Type	Length	Constraint
manufactID	PK			NUMBER	10	PRIMARY KEY
manufactName				VARCHAR2	30	NOT NULL
manufactAddress				VARCHAR2	50	NOT NULL
manufactPcode				VARCHAR2	8	NOT NULL
manufactTel				VARCHAR2	11	
manufactEmail				VARCHAR2	30	UNIQUE, NOT NULL
manufactMob				VARCHAR2	11	
manufactWeb				VARCHAR2	30	

Data Dictionary: manufact Link

Attribute Name	Key Type	FK Table	FK Column	Data Type	Length	Constraint
manufactID	PK FK	manufacturer_record	manufactID	NUMBER	10	PRIMARY KEY , FOREIGN KEY
partNo	PK FK	part_record	partNo	NUMBER	10	PRIMARY KEY , FOREIGN KEY

Data Dictionary: class_record

Attribute Name	Key Type	FK Table	FK Column	Data Type	Length	Constraint
classID	PK			NUMBER	10	PRIMARY KEY
classType				VARCHAR2	50	NOT NULL
fullDayRate				DECIMAL(7,2)	7	NOT NULL
HalfDayRate				DECIMAL(7,2)	7	NOT NULL

Data Dictionary: order_record

Attribute Name	Key Type	FK Table	FK Column	Data Type	Length	Constraint
orderNo	PK			NUMBER	10	PRIMARY KEY
orderDate				DATE	dd-mm-yyyy	NOT NULL
subtotal				NUMBER	(7,2)	
VAT				NUMBER	(7,2)	
total				NUMBER	(7,2)	

Data Dictionary: parts_record

Attribute Name	Key Type	FK Table	FK Column	Data Type	Length	Constraint
partsNo	PK			NUMBER	10	PRIMARY KEY
partname				VARCHAR2	20	NOT NULL
price				NUMBER	(7,2)	NOT NULL

Data Dictionary: partsOrder

Attribute Name	Key Type	FK Table	FK Column	Data Type	Length	Constraint
qty				NUMBER	4	NOT NULL
lineTotal				NUMBER	(7,2)	
orderNo	PK FK			NUMBER	10	PRIMARY KEY , FOREIGN KEY
partsNo	PK FK			NUMBER	10	PRIMARY KEY , FOREIGN KEY

Data Dictionary: sold_bikes

Attribute Name	Key Type	FK Table	FK Column	Data Type	Length	Constraint
bikeID	PK FK			NUMBER	10	PRIMARY KEY , FOREIGN KEY
dlrID				NUMBER	10	PRIMARY KEY , FOREIGN KEY
saleDate				DATE	dd-mm-yyyy	
salePrice				NUMBER	(7,2)	(salePrice > 0)

17.0 SQL Documentation

A screen shot to show the drop sequences working is shown in Figure 9.0

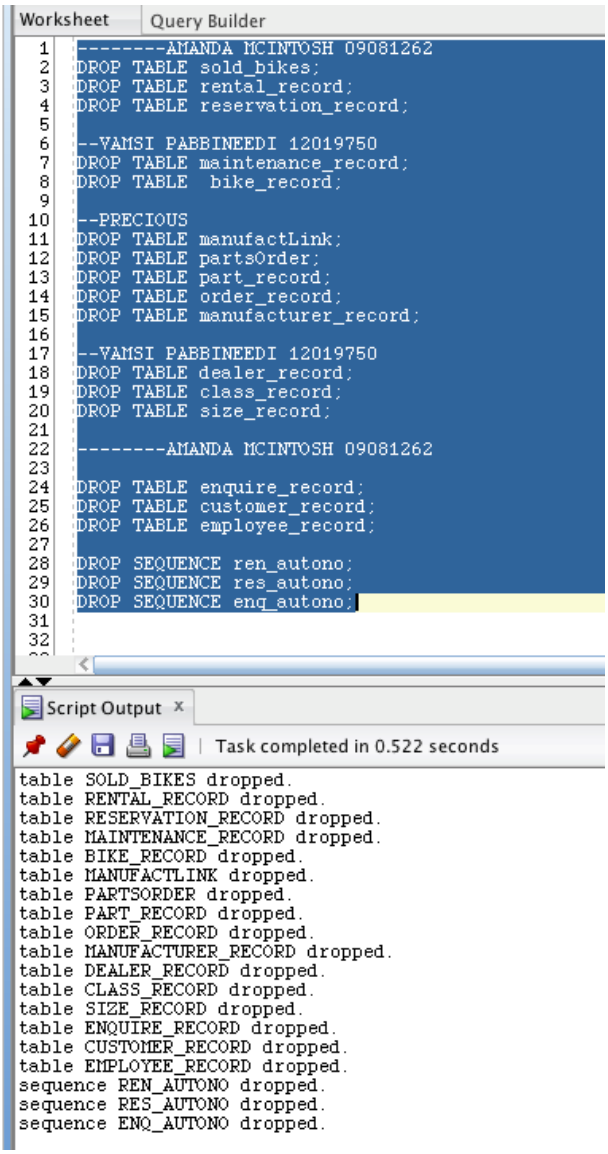


Figure 9.0

Information Systems Project

The working create statements are shown in the figure 9.1 and 9.2 screenshots.

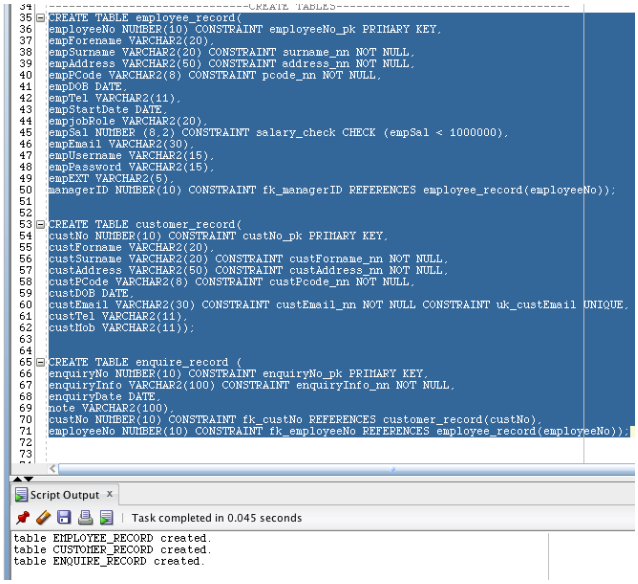


Figure 9.1

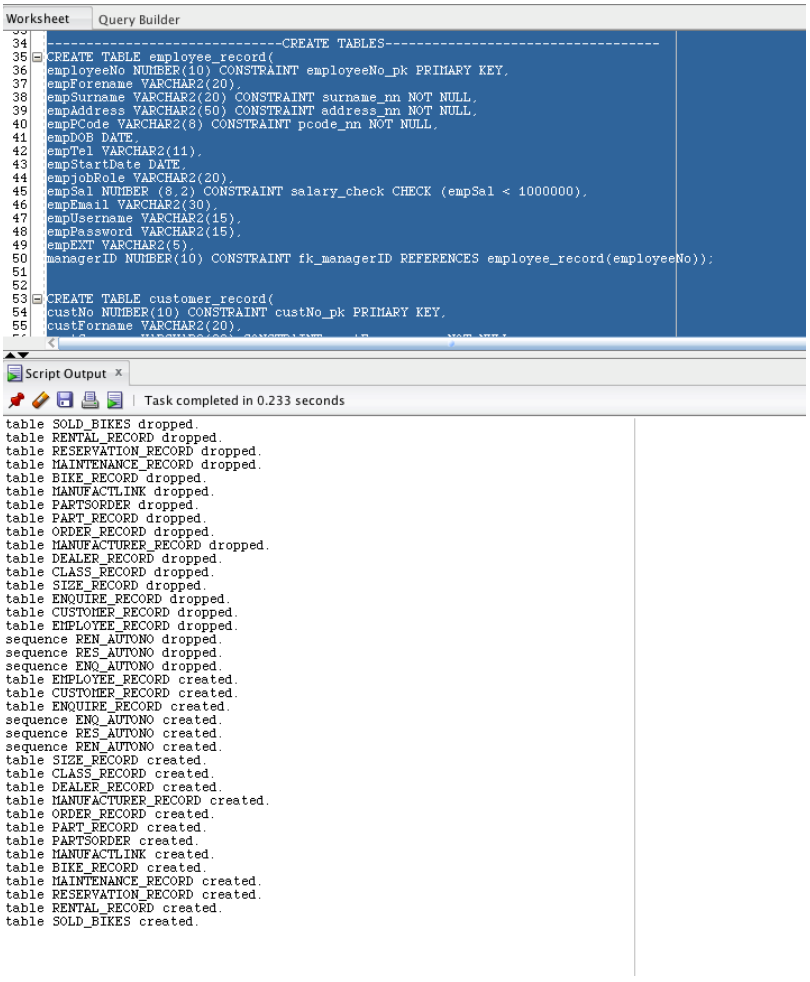


Figure 9.2

The working insert statements are shown in figure 9.3.

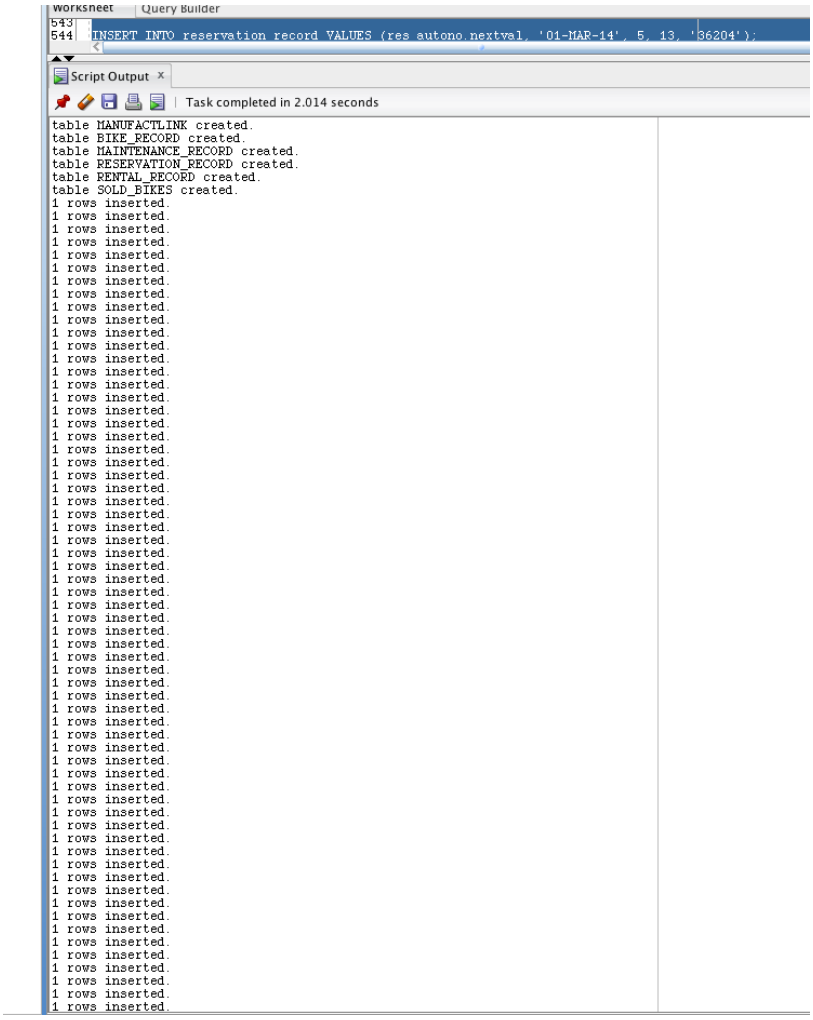
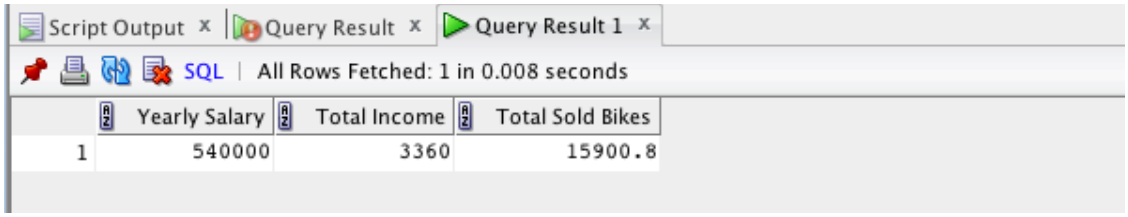


Figure 9.3

17.1 Amanda McIntosh's SQL Queries

Figure 10.0 shows a query for total salary paid in a year, the total rent paid and total sold bikes. This is calculated using the SUM function. This query gives an overview of the businesses' outgoings and income and could aid Ray in financial decisions.

```
SELECT SUM(empSal) "Yearly Salary", SUM(rentPaid) "Total Income", SUM(salePrice)"Total Sold Bikes"

FROM employee_record

NATURAL JOIN  rental_record, sold_bikes;
```

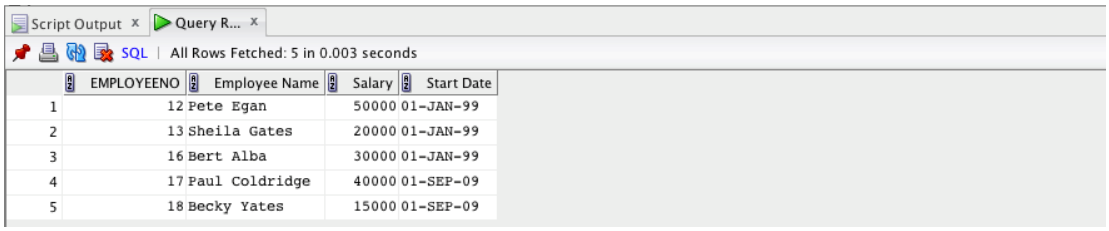


	Yearly Salary	Total Income	Total Sold Bikes
1	540000	3360	15900.8

Figure 10.0

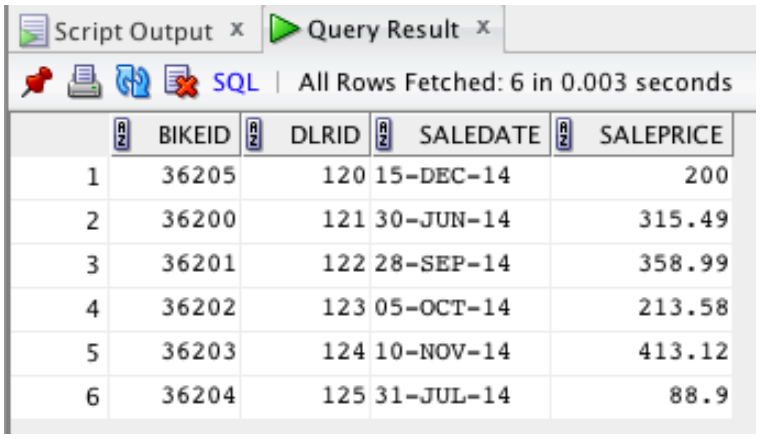
In figure 10.1 an SQL query displays an employee's number and full name when their salary is greater than 10, 000 and they started working for Rays Rentals before January 2010. The concatenation operator was used to combine the employee forename and surname and a column alias has been provided. The WHERE clause was used to select employees earning a higher salary than 10,000 and the AND Logical operator was used to specify the start date required. This query could be used by Ray's Rentals to determine who is entitled to a pay rise.

```
SELECT employeeNo, empForename|| ' '|| empSurname AS "Employee Name", empSal "Salary", empStartDate
"Start Date"

FROM employee_record

WHERE empSal > 10000

AND empStartDate < '01-JAN-10';
```



	EMPLOYEENO	Employee Name	Salary	Start Date
1	12	Pete Egan	50000	01-JAN-99
2	13	Sheila Gates	20000	01-JAN-99
3	16	Bert Alba	30000	01-JAN-99
4	17	Paul Coldridge	40000	01-SEP-09
5	18	Becky Yates	15000	01-SEP-09

Figure 10.1



Script Output | Query Result

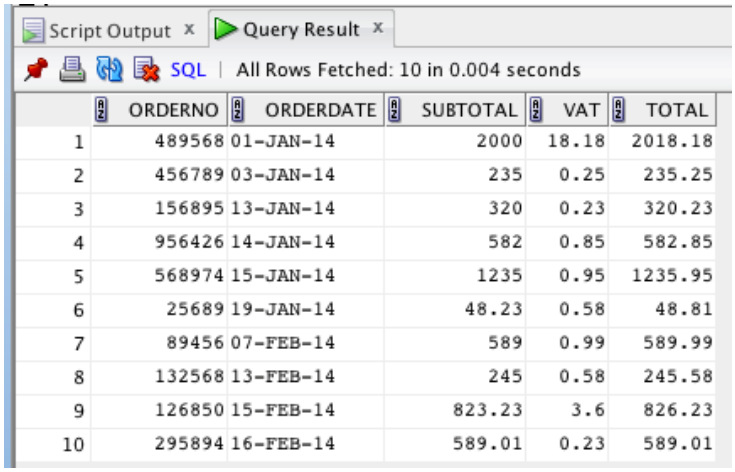
SQL | All Rows Fetched: 6 in 0.003 seconds

	BIKEID	DLRID	SALEDATE	SALEPRICE
1	36205	120	15-DEC-14	200
2	36200	121	30-JUN-14	315.49
3	36201	122	28-SEP-14	358.99
4	36202	123	05-OCT-14	213.58
5	36203	124	10-NOV-14	413.12
6	36204	125	31-JUL-14	88.9

Figure 10.2

A SELECT ALL (*) statement has been used to display all of the sold bike records in figure 10.2. This can give Ray an overview of the income gained from decommissioned bike sales.

```
SELECT *

FROM sold_bikes;
```



Script Output | Query Result

SQL | All Rows Fetched: 10 in 0.004 seconds

	ORDERNO	ORDERDATE	SUBTOTAL	VAT	TOTAL
1	489568	01-JAN-14	2000	18.18	2018.18
2	456789	03-JAN-14	235	0.25	235.25
3	156895	13-JAN-14	320	0.23	320.23
4	956426	14-JAN-14	582	0.85	582.85
5	568974	15-JAN-14	1235	0.95	1235.95
6	25689	19-JAN-14	48.23	0.58	48.81
7	89456	07-FEB-14	589	0.99	589.99
8	132568	13-FEB-14	245	0.58	245.58
9	126850	15-FEB-14	823.23	3.6	826.23
10	295894	16-FEB-14	589.01	0.23	589.01

Figure 10.3

Figure 10.3 uses the SELECT ALL (*) statement to display all of the order records and the ORDER BY clause is used to order the results in ascending order by the order date. This query could enable estimations of the monthly order costs.

```
SELECT *

FROM order_record

ORDER BY orderDate ASC;
```

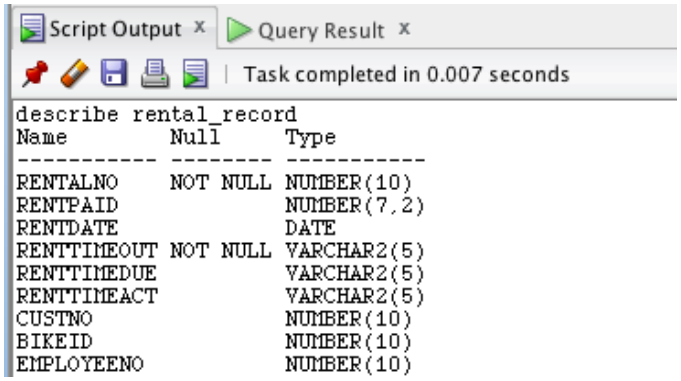


Figure 10.4

A describe command to display the table information for rental record has been illustrated in figure 10.4. This helps in the creation of queries and can aid in data input.

describe rental_record

In figure 10.5 the SUM function has been used to calculate the total rent paid up to today's date using the sysdate function. This gives ray a running total of the businesses main income.

SELECT SUM(rentPaid) "Income"

FROM rental_record

WHERE rentDate < sysdate;

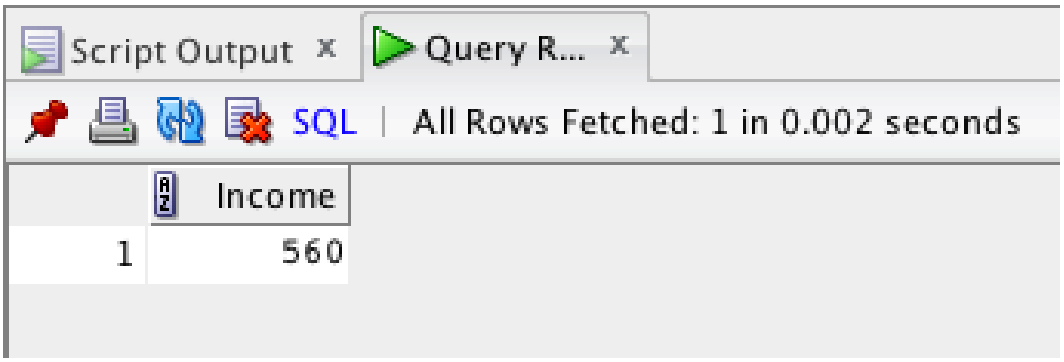


Figure 10.5

In figure 10.6 the group function has been used to list the sum of rent paid by month. The to_char function has been used to make the group by month possible. This shows Ray his busiest months and could help with forward planning.

```
SELECT to_char(rentDate,'MONTH YYYY')"Month", SUM(rentPaid)"Rent Paid"

FROM rental_record

GROUP BY to_char(rentDate,'MONTH YYYY');
```

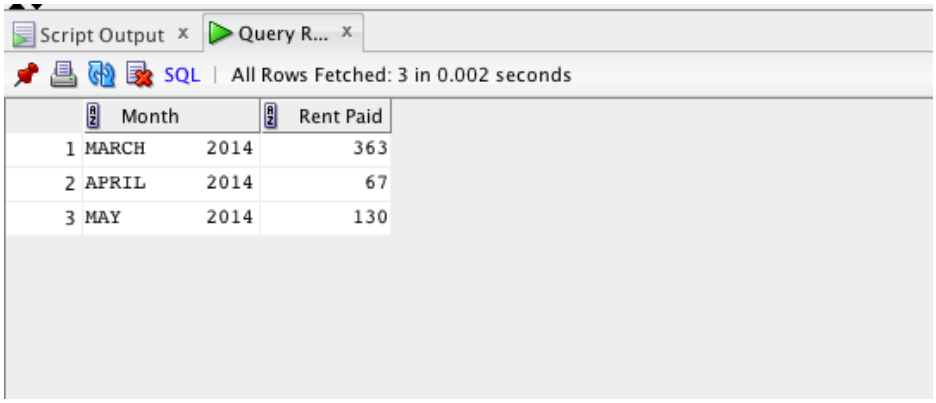


Figure 10.6

Figure 10.7 shows the group function being used to list the sum of orders by month. The to_char function has been used to make the group by month possible. This could help with Ray's financial planning.

```
SELECT to_char(orderDate,'MONTH YYYY')"Month", SUM(total)"Outgoings"

FROM order_record

GROUP BY to_char(orderDate,'MONTH YYYY');
```

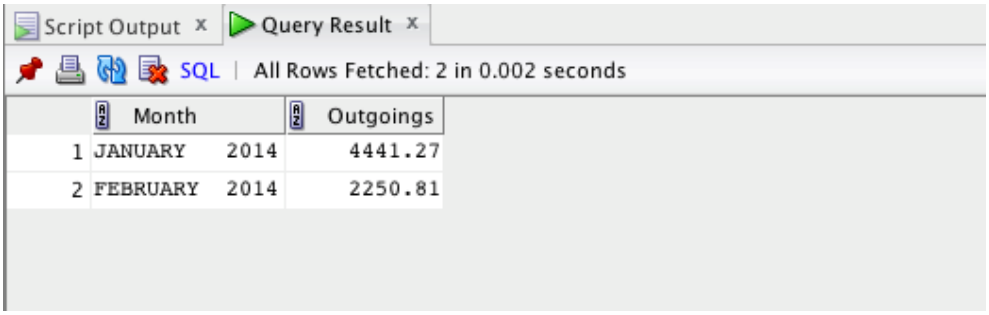
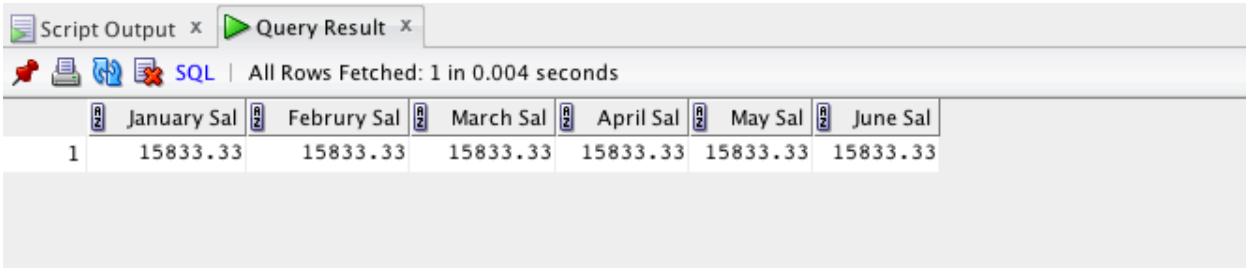


Figure 10.7

The monthly salary has been calculated in figure 10.8; using the SUM function and the divide arithmetic operator. The calculation has been rounded to two decimal places using the round function. The salary for months, January to June have been displayed using an alias of the month name.

```
SELECT ROUND((SUM(empSal)/12),2) "January Sal", ROUND((SUM(empSal)/12),2) "Februry Sal",

ROUND((SUM(empSal)/12),2) "March Sal", ROUND((SUM(empSal)/12),2) "April Sal",

ROUND((SUM(empSal)/12),2) "May Sal",

 ROUND((SUM(empSal)/12),2) "June Sal"

FROM employee_record;
```



	January Sal	Februry Sal	March Sal	April Sal	May Sal	June Sal
1	15833.33	15833.33	15833.33	15833.33	15833.33	15833.33

Figure 10.8

4.2 Vamsi Pabbineedi 's SQL Queries

SQL Query Results for Exception Management Report – Maintenance History

Figure 11.0 shows a basic SELECT statement that identifies the maintenance columns to be displayed from the maintenance record table.

```
-- Default table display

SELECT maintenanceid, maintenancedate,faultdesc,faultdate,actiontaken,actiontakendate

FROM maintenance_record;
```

	MAINTENANCEID	MAINTENANCEDATE	FAULTDESC	FAULTDATE	ACTIONTAKEN	ACTIONTAKENDATE
1	1	15-FEB-14	Puncture-rear	15-FEB-14	New Inner Tube	16-FEB-14
2	2	10-MAR-14	Chain keeps breaking	10-MAR-14	Fit New Chain	27-AUG-14
3	3	08-FEB-14	Not Applicable	08-FEB-14	NormalService	01-MAR-14
4	4	22-MAR-14	Not Applicable	22-MAR-14	Normal Service	03-MAR-14
5	5	14-FEB-14	Split Saddle	14-FEB-14	Fit new saddle	15-FEB-14
6	6	17-JAN-14	Faulty brakes	17-JAN-14	New brakes fitted	15-FEB-14

Figure 11.0

Figure 11.1 illustrates an SQL query involving a column alias and concatenation operator.

```
--Column Alias/Concatenation Operator

SELECT faultdesc|| ' '||maintenanceid AS "Bike Fault Type"

FROM maintenance_record;
```

	Bike Fault Type
1	Puncture-rear 1
2	Chain keeps breaking 2
3	Not Applicable 3
4	Not Applicable 4
5	Split Saddle 5
6	Faulty brakes 6

Figure 11.1

In Figure 11.2 a DESCRIBE command was used to display information about the structure of the maintenance record table.

--Describe command

DESCRIBE maintenance_record

```
DESCRIBE maintenance_record
Name            Null     Type
--------------- -------- ------------
MAINTENANCEID   NOT NULL NUMBER(10)
MAINTENANCEDATE NOT NULL DATE
FAULTDESC                VARCHAR2(50)
FAULTDATE       NOT NULL DATE
ACTIONTAKENDATE          DATE
ACTIONTAKEN     NOT NULL VARCHAR2(30)
EMPLOYEENO               NUMBER(10)
BIKEID                   NUMBER(10)
```

Figure 11.2

Figure 11.3 shows the WHERE clause , a SQL query that limits or restricts the type of data required(displayed).

--Where clause

SELECT maintenanceid, maintenancedate,faultdesc,faultdate,actiontaken,actiontakendate

FROM maintenance_record

WHERE maintenanceid = 4;

	MAINTENANCEID	MAINTENANCEDATE	FAULTDESC	FAULTDATE	ACTIONTAKEN	ACTIONTAKENDATE
1	4	22-MAR-14	Not Applicable	22-MAR-14	Normal Service	03-MAR-14

Figure 11.3

Figures 11.4 and 11.5 also use the WHERE clause, but is a SQL query that identifies or retrieves specific character strings and dates

--Character Strings and Dates

SELECT maintenanceid,faultdesc

FROM maintenance_record

WHERE faultdate = '15-FEB-14';

	MAINTENANCEID	FAULTDESC
1	1	Puncture-rear

Figure 11.4

```
SELECT maintenanceid,maintenancedate,faultdate,actiontaken,actiontakendate

FROM maintenance_record

WHERE faultdesc = 'Chain keeps breaking';
```

	MAINTENANCEID	MAINTENANCEDATE	FAULTDATE	ACTIONTAKEN	ACTIONTAKENDATE
1	2	10-MAR-14	10-MAR-14	Fit New Chain	27-AUG-14

Figure 11.5

Figures 11.7 and 11.8 highlights a Comparison operator that retrieves data that is from a specific range or parameter i.e. BETWEEN “maintenanceid” – “2&6”.

```
--Comparison operators

SELECT maintenanceid,maintenancedate,faultdesc ,faultdate,actiontaken,actiontakendate

FROM maintenance_record

WHERE actiontakendate < '3-MAR-14';
```

	MAINTENANCEID	MAINTENANCEDATE	FAULTDESC	FAULTDATE	ACTIONTAKEN	ACTIONTAKENDATE
1	1	15-FEB-14	Puncture-rear	15-FEB-14	New Inner Tube	16-FEB-14
2	3	08-FEB-14	Not Applicable	08-FEB-14	NormalService	01-MAR-14
3	5	14-FEB-14	Split Saddle	14-FEB-14	Fit new saddle	15-FEB-14
4	6	17-JAN-14	Faulty brakes	17-JAN-14	New brakes fitted	15-FEB-14

Figure 11.7

```
SELECT maintenanceid, maintenancedate,faultdesc,faultdate,actiontaken,actiontakendate

FROM maintenance_record

WHERE maintenanceid  BETWEEN 2 AND 5 ;
```

	MAINTENANCEID	MAINTENANCEDATE	FAULTDESC	FAULTDATE	ACTIONTAKEN	ACTIONTAKENDATE
1	2	10-MAR-14	Chain keeps breaking	10-MAR-14	Fit New Chain	27-AUG-14
2	3	08-FEB-14	Not Applicable	08-FEB-14	NormalService	01-MAR-14
3	4	22-MAR-14	Not Applicable	22-MAR-14	Normal Service	03-MAR-14
4	5	14-FEB-14	Split Saddle	14-FEB-14	Fit new saddle	15-FEB-14

Figure 11.8

Figure 11.9 is a Join of two tables involving the USING clause.

```
SELECT maintenanceid, maintenancedate,faultdesc,faultdate,actiontaken ,bikeID ,bikeMake,bikeModel

FROM maintenance_record

JOIN bike_record

USING (bikeID);
```

	MAINTENANCEID	MAINTENANCEDATE	FAULTDESC	FAULTDATE	ACTIONTAKEN	BIKEID	BIKEMAKE	BIKEMODEL
1	1	15-FEB-14	Puncture-rear	15-FEB-14	New Inner Tube	36200	Superbike	Explorer
2	2	10-MAR-14	Chain keeps breaking	10-MAR-14	Fit New Chain	36201	Atom	Speedster
3	3	08-FEB-14	Not Applicable	08-FEB-14	NormalService	36202	Razer	Sharp
4	4	22-MAR-14	Not Applicable	22-MAR-14	Normal Service	36203	Knight	Excel
5	5	14-FEB-14	Split Saddle	14-FEB-14	Fit new saddle	36204	Neo	Ritz
6	6	17-JAN-14	Faulty brakes	17-JAN-14	New brakes fitted	36205	Voltech	Spark

Figure 11.9

Figure 12.0 is Using Case Conversion - Displays maintenance table columns for string “normal service” from attribute “actiontaken”.

```
--Using Case-Conversion

SELECT maintenanceid, maintenancedate,faultdesc,faultdate,actiontaken,actiontakendate

FROM maintenance_record

WHERE LOWER(actiontaken) LIKE 'normalservice';
```

	MAINTENANCEID	MAINTENANCEDATE	FAULTDESC	FAULTDATE	ACTIONTAKEN	ACTIONTAKENDATE
1	3	08-FEB-14	Not Applicable	08-FEB-14	NormalService	01-MAR-14

Figure 12.0

Figure 12.1 -The Order by clause query. The data sets in maintenance history tables are in order of 'actiontakendates'(from the earliest date to last known date).

--Order by clause

SELECT maintenanceid, maintenancedate,faultdesc,faultdate,actiontaken,actiontakendate

FROM maintenance_record

ORDER BY actiontakendate;

	MAINTENANCEID	MAINTENANCEDATE	FAULTDESC	FAULTDATE	ACTIONTAKEN	ACTIONTAKENDATE
1	5	14-FEB-14	Split Saddle	14-FEB-14	Fit new saddle	15-FEB-14
2	6	17-JAN-14	Faulty brakes	17-JAN-14	New brakes fitted	15-FEB-14
3	1	15-FEB-14	Puncture-rear	15-FEB-14	New Inner Tube	16-FEB-14
4	3	08-FEB-14	Not Applicable	08-FEB-14	NormalService	01-MAR-14
5	4	22-MAR-14	Not Applicable	22-MAR-14	Normal Service	03-MAR-14
6	2	10-MAR-14	Chain keeps breaking	10-MAR-14	Fit New Chain	27-AUG-14

Figure 12.1

Figure 12.2 shows the use Naturals joins of three tables. Combining data of other tables that might be relevant to Ray's Rentals maintenance history.

--Natural joins of three tables

SELECT maintenanceid, maintenancedate,faultdesc,faultdate,actiontaken ,bikeID ,bikeMake,bikeModel ,sizeid,sizetype,classid,classtype

FROM maintenance_record

NATURAL JOIN bike_record

NATURAL JOIN size_record

NATURAL JOIN class_record;

MAINTENANCEID	MAINTENANCEDATE	FAULTDESC	FAULTDATE	ACTIONTAKEN	ACTIONTAKENDATE	BIKEID	BIKEMAKE	BIKEMODEL	SIZEID	SIZETYPE	CLASSID	CLASSTYPE
5	14-FEB-14	Split Saddle	14-FEB-14	Fit new saddle	15-FEB-14	36204	Neo	Ritz	104	child	112	tandem
6	17-JAN-14	Faulty brakes	17-JAN-14	New brakes fitted	15-FEB-14	36205	Voltech	Spark	102	small male	111	road
1	15-FEB-14	Puncture-rear	15-FEB-14	New Inner Tube	16-FEB-14	36200	Superbike	Explorer	100	large male	110	mountain
3	08-FEB-14	Not Applicable	08-FEB-14	NormalService	01-MAR-14	36202	Razer	Sharp	102	small male	112	tandem
4	22-MAR-14	Not Applicable	22-MAR-14	Normal Service	03-MAR-14	36203	Knight	Excel	103	standard female	110	mountain
2	10-MAR-14	Chain keeps breaking	10-MAR-14	Fit New Chain	27-AUG-14	36201	Atom	Speedster	101	standard male	111	road

Figure 12.2

Figure 12.3 -SQL query involving an arithmetic operator with dates. Calculates the number of weeks by subtracting the 'actiontakendate' from the current date (SYSDATE) and dividing by 7 .

```
--Arithmetic operator with dates

SELECT faultdate, (SYSDATE-actiontakendate)/7 AS WEEKS

FROM maintenance_record

WHERE maintenanceid = 2;
```

	FAULTDATE	WEEKS
1	10-MAR-14	-23.92546957671957671957671957671957671957

Figure 12.3

Figure 12.4 -The following figure illustrates an SQL query that combines Naturals joins of three tables and an Order by function. The data sets are displayed or retrieved in ascending order of 'actiontakendate'.

```
--Order

SELECT maintenanceid, maintenancedate,faultdesc,faultdate,actiontaken,actiontakendate ,bikeID
,bikeMake,bikeModel ,sizeid,sizetype,classid,classtype

FROM maintenance_record

NATURAL JOIN bike_record

NATURAL JOIN size_record

NATURAL JOIN class_record

ORDER BY actiontakendate ;
```

MAINTENANCEID	MAINTENANCEDATE	FAULTDESC	FAULTDATE	ACTIONTAKEN	ACTIONTAKENDATE	BIKEID	BIKEMAKE	BIKEMODEL	SIZEID	SIZETYPE	CLASSID	CLASSTYPE
5	14-FEB-14	Split Saddle	14-FEB-14	Fit new saddle	15-FEB-14	36204	Neo	Ritz	104	child	112	tandem
6	17-JAN-14	Faulty brakes	17-JAN-14	New brakes fitted	15-FEB-14	36205	Voltech	Spark	102	small male	111	road
1	15-FEB-14	Puncture-rear	15-FEB-14	New Inner Tube	16-FEB-14	36200	Superbike	Explorer	100	large male	110	mountain
3	08-FEB-14	Not Applicable	08-FEB-14	NormalService	01-MAR-14	36202	Razer	Sharp	102	small male	112	tandem
4	22-MAR-14	Not Applicable	22-MAR-14	Normal Service	03-MAR-14	36203	Knight	Excel	103	standard female	110	mountain
2	10-MAR-14	Chain keeps breaking	10-MAR-14	Fit New Chain	27-AUG-14	36201	Atom	Speedster	101	standard male	111	road

Figure 12.4

17.3 Precious Igbinosun's SQL Queries

```
SELECT  bikeid, manufactid, manufactname,  bikemodel, sizeid, sizetype, dlrid, dlrname, maintenanceid,
ROUND(MONTHS_BETWEEN(SYSDATE,purchasedate),0) AS AGE_OF_BIKE_IN_MONTHS
FROM bike_record
NATURAL JOIN size_record
NATURAL JOIN dealer_record
NATURAL JOIN manufacturer_record
NATURAL JOIN maintenance_record
WHERE MONTHS_BETWEEN (SYSDATE, purchasedate) >=12;
```

This select statement (figure 13.0) allows users to search for the history of each bike, which includes the age of bikes in months, model and manufacturer, bike number after it was sold to dealers. This will allow Ray's Rentals to give an approximated date when bikes are been sold out.

Query Result

SQL | All Rows Fetched: 30 in 0.008 seconds

	BIKEID	MANUFACTID	MANUFACTNAME	BIKEMODEL	SIZEID	SIZETYPE	DLRID	DLRNAME	MAINTENANCEID	AGE_OF_BIKE_IN_MONTHS
1	36200	1853	Techcorporation	Explorer	100	large male	120	JS Bikes	1	13
2	36200	1853	Techcorporation	Explorer	100	large male	121	Dean Jone Peaks	1	13
3	36200	1853	Techcorporation	Explorer	100	large male	122	Walker Rides	1	13
4	36200	1853	Techcorporation	Explorer	100	large male	123	Debbie Wheels	1	13
5	36200	1853	Techcorporation	Explorer	100	large male	124	Terrain Cycles	1	13
6	36200	1853	Techcorporation	Explorer	100	large male	125	Nolan Bike Rentals	1	13
7	36201	20811	Stimholdings	Speedster	101	standard male	120	JS Bikes	2	14
8	36201	20811	Stimholdings	Speedster	101	standard male	121	Dean Jone Peaks	2	14
9	36201	20811	Stimholdings	Speedster	101	standard male	122	Walker Rides	2	14
10	36201	20811	Stimholdings	Speedster	101	standard male	123	Debbie Wheels	2	14
11	36201	20811	Stimholdings	Speedster	101	standard male	124	Terrain Cycles	2	14
12	36201	20811	Stimholdings	Speedster	101	standard male	125	Nolan Bike Rentals	2	14
13	36202	1549	sundex	Sharp	102	small male	120	JS Bikes	3	14
14	36202	1549	sundex	Sharp	102	small male	121	Dean Jone Peaks	3	14
15	36202	1549	sundex	Sharp	102	small male	122	Walker Rides	3	14
16	36202	1549	sundex	Sharp	102	small male	123	Debbie Wheels	3	14
17	36202	1549	sundex	Sharp	102	small male	124	Terrain Cycles	3	14
18	36202	1549	sundex	Sharp	102	small male	125	Nolan Bike Rentals	3	14
19	36203	70284	Techcan	Excel	103	standard female	120	JS Bikes	4	13
20	36203	70284	Techcan	Excel	103	standard female	121	Dean Jone Peaks	4	13
21	36203	70284	Techcan	Excel	103	standard female	122	Walker Rides	4	13
22	36203	70284	Techcan	Excel	103	standard female	123	Debbie Wheels	4	13
23	36203	70284	Techcan	Excel	103	standard female	124	Terrain Cycles	4	13
24	36203	70284	Techcan	Excel	103	standard female	125	Nolan Bike Rentals	4	13
25	36204	9586	Kontouch	Ritz	104	child	120	JS Bikes	5	12
26	36204	9586	Kontouch	Ritz	104	child	121	Dean Jone Peaks	5	12
27	36204	9586	Kontouch	Ritz	104	child	122	Walker Rides	5	12
28	36204	9586	Kontouch	Ritz	104	child	123	Debbie Wheels	5	12
29	36204	9586	Kontouch	Ritz	104	child	124	Terrain Cycles	5	12
30	36204	9586	Kontouch	Ritz	104	child	125	Nolan Bike Rentals	5	12

Figure 13.0

```
SELECT dlrname AS Dealer_Name, bikeid, sizeid, sizetype, purchaseprice, saleprice, ROUND(saleprice -
purchaseprice)AS Profit_From_Bike_Sales
FROM DEALER_RECORD
NATURAL JOIN bike_record
NATURAL JOIN sold_bikes
NATURAL JOIN SIZE_RECORD;
```

This report (figure 13.1) show the profit made on a bike after it been sold, it shows the buyer's name, bikeID, the sizeID and type

Script Output | Query Result

SQL | All Rows Fetched: 6 in 0.015 seconds

	DEALER_NAME	BIKEID	SIZEID	SIZETYPE	PURCHASEPRICE	SALEPRICE	PROFIT_FROM_BIKE_SALES
1	JS Bikes	36205	102	small male	220	200	-20
2	Dean Jone Peaks	36200	100	large male	350	315.49	-35
3	Walker Rides	36201	101	standard male	400	358.99	-41
4	Debbie Wheels	36202	102	small male	256	213.58	-42
5	Terrain Cycles	36203	103	standard female	480	413.12	-67
6	Nolan Bike Rentals	36204	104	child	157	88.9	-68

Figure 13.1

```
SELECT orderno, orderdate, partno, 3*price AS extra_order, ((3*price) + total) AS new_total
FROM partsorder
NATURAL JOIN order_record
NATURAL JOIN part_record
WHERE orderdate LIKE '%JAN%'
ORDER BY orderno;
```

In January instead of placing new order for extra part, additional parts was order and added to an existing order and a new total is displayed with this query showing the orderno it was added to, the date the order was placed, the partno and the subtotal for the extra.

Script Output | Query...

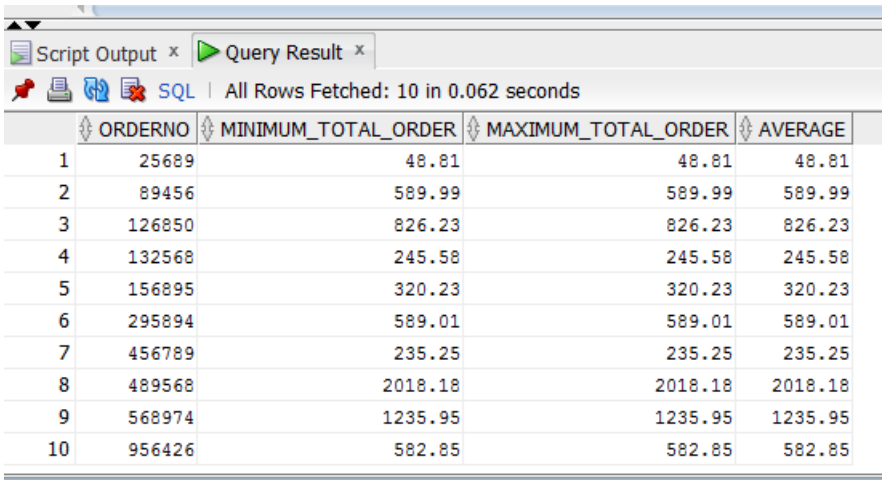
SQL | All Rows Fetched: 9 in 0.016 seconds

	ORDERNO	ORDERDATE	PARTNO	EXTRA_ORDER	NEW_TOTAL
1	25689	19-JAN-14	1005	66.06	114.87
2	156895	13-JAN-14	1005	66.06	386.29
3	456789	03-JAN-14	199	298.68	533.93
4	489568	01-JAN-14	199	298.68	2316.86
5	489568	01-JAN-14	985	135.06	2153.24
6	489568	01-JAN-14	178	106.47	2124.65
7	568974	15-JAN-14	756	116.76	1352.71
8	956426	14-JAN-14	426	135.75	718.6
9	956426	14-JAN-14	756	116.76	699.61

Figure 13.2

```
SELECT orderno, MIN(total) AS Minimum_total_order, MAX(total) AS Maximum_total_order,  AVG(total) AS
Average
FROM order_record
GROUP BY orderno;
```

This screenshot below (figure 13.3) shows the maximum, minimum and average of the total cost of every order made.


Script Output | Query Result

SQL | All Rows Fetched: 10 in 0.062 seconds

	ORDERNO	MINIMUM_TOTAL_ORDER	MAXIMUM_TOTAL_ORDER	AVERAGE
1	25689	48.81	48.81	48.81
2	89456	589.99	589.99	589.99
3	126850	826.23	826.23	826.23
4	132568	245.58	245.58	245.58
5	156895	320.23	320.23	320.23
6	295894	589.01	589.01	589.01
7	456789	235.25	235.25	235.25
8	489568	2018.18	2018.18	2018.18
9	568974	1235.95	1235.95	1235.95
10	956426	582.85	582.85	582.85

Figure 13.3

18.0 Commentary

18.1 Amanda McIntosh

When reviewing the ERD it was apparent that the addition of certain attributes would be necessary for the management reports, for example, the action taken date was needed for the Exception Report and the employee salary was needed for the Analysis Report. Due to the original Analysis Report being substantially elaborate, the report was broken down into separate sections. The monthly salary was calculated by dividing the sum of the total salaries by 12. The sum of rent paid and order total were grouped by month. The importance of the foreign and primary keys became apparent when linking tables, natural joins were ideal when linking all of the data from numerous tables, however when linking tuples the primary and foreign keys are essential.

When deciding the data types for telephone numbers, Varchar2 was chosen to avoid the number data type eliminating the first 0 from telephone numbers. Sequences were created for each primary key to provide each record with a unique identifier, however, this caused difficulties when adding insert statements (as each record would need to be inserted separately). To prevent these difficulties the sequences were only created for rentalNo, enquiryNo and resNo . The naming of tables and attribute were more important than initially thought, some entities were named using SQL syntax which created run errors. It also became evident that each entity name and constraint named needed to be unique.

18.2 Vamsi Pabbineedi

As a team we had to ensure that the ERDs used contained suitable data types and sizes. While amending the ERDs it became clear that additional attributes were required to create the management reports i.e. “actiontakendate” for exception management report (maintenance history). It was also important that our primary and foreign keys constraints were used in relation to the ERDs. Furthermore we had to ensure that all populated data from the tables were relevant to the Rays Rental case study as well the management reports outlined in part one. While implementing the oracle database, SQL queries were used to create specific management reports each of which have distinct purposes. For instance I used SQL queries to implement a maintenance history report (exception management report). In a real-world situation this exception report is designed to help Ray’s Rentals identify cases that generally require attention. However due to minor constraints, I was unable to provide this type of facility. As for the SQL queries, I have used columns aliases, comparison operators, natural joins and many more to implement the management report.

18.2 Precious Igbinosun

While creating the database, it was clear that we needed to add another entity to the ERD to eliminate null values in the table. So this new entity was called “Sold Bike”, basically acts as an archive for all the sold bikes. It has a one to one relationship with the bike record and a one to many relationship with the dealer’s record. Other changes were made too, like the data type for date and telephone are varchar. The queries demonstrates a good understanding of SQL systax such as natural join, column alias, comparison and logical operation, order and group by functions. Also, composite key was implemented in the data dictionary and in creating the SQL table. Personally, I have been able to write a complex report joining more than two tables. The use of sysdate in my report, helped to calculate the age of each bike in months.

19.0 Project Conclusion

19.1 Amanda's Project Conclusion

Thorough the progression of this assignment the importance of a design and development process has been validated. Design processes are dynamic and require constant revisions at each stage in order to produce the greatest obtainable database; this highlights the importance of a good team relationship and communications skills.

This development process helped to establish that a computer based system is vital for the future of Ray's Rentals, it will not only help to analyse the financial stability of the business but will vastly improve customer relationships. Other shortcomings of the current paper based system, which would be resolved, include: data security, integrity and storage. The most suitable option for this business would be a stand alone system as this would reduce costs and the amount of technical support needed. With the use of management reports and having a secure, reliable way to save necessary data, business processes will become less arduous and more efficient.

Careful scrutiny of the case study and analysing the current paper based forms helped to establish the basic structure the database may have need. Use case diagrams help to provide an overall look at what happens in the business being evaluated; basic processes can then be considered further by producing use case specifications. SQL data types and the naming of tables and exceptions are vital, simple names and following a structured naming format enabled the code and queries to be written quickly.

19.2 Vamsi's Project Conclusion

Undertaking this information systems project allowed me to gain an insight into what project management in IT (or software/systems development) might involve. Despite the small scale of the project, it still required the efforts of team work to ensure the project was completed successfully to an exceptional standard .As discovered during this project, communication skills became vital especially when trying to share or collaborate ideas and meet deadlines on a monthly basis. Also it became apparent while carrying out a project in a team, the delegation of tasks and effective leadership was necessary for retaining a smooth project workflow.

Furthermore, this project taught me aspects of the systems development lifecycle. In the context of Ray's Rentals, as a team we successfully implemented a computerised database system that would help resolve issues of a paper-based system previously used by Ray. This wouldn't have been possible without the following key phases of the project; feasibility study, SA&D (systems analysis and design) and testing. Most importantly, these key phases of the project enabled the team to outline problems, suggest solutions, model business environments by using UCDs(use case diagrams) and define database structures by using ERDs(Entity Relationship Diagrams).This collectively as a whole allowed for a successful implementation of an information system.

19.3 Precious's Project Conclusion

Working in a group has improved the communication skills between every member of team and I have learnt about management system. We started this project by looking into the various types of management reports, which was used to summarise the problems faced by Ray's Rentals. A use case diagram was used to visualise the interaction between the system and it users (actors) with the requirements. The next step was to create use case specification for each process in the UCD. During the creations of the ERD (entity relationship diagram), I was able to assign primary and foreign keys to attributes, which defines an entity. To conclude, the final model of this project was produced based on the fact that all stages of producing a system was followed and this a functioning user-friendly database that provide a full solution to the business.

20.0 References

Carbongen Amcis. (No date) *Project Management.* [Online Image] [Accessed on 15TH October 3013]

Carbogen Amcis, (online) , "Project Management" (cited 15th October2013) , Available from <http://www.carbogen-amcis.com/services/project-management.asp>

Chirantan Basu (online),"What Are the Basic Kinds of Reports Produced by an Management Information System" (cited 15th October 2013), Available from <http://smallbusiness.chron.com/basic-kinds-reports-produced-management-information-system-30845.html>

Expert Market, (online), "What is an EPOS system and How Does it Work" (cited 21st October 2013), Available from <http://epos.expertmarket.co.uk/what-is-an-epos-system-and-how-does-it-work>

InetSoft, (online), "Ad Hoc Analysis" (cited on 20th October 2013), Available from <http://www.inetsoft.com/info/ad_hoc_olap/>

Margaret Rouse, (online), 'Ad Hoc Analysis' (cited 21st October 2013), Available from <http://searchbusinessanalytics.techtarget.com/definition/ad-hoc-analysis>)

QFINANCE,(online),"Definition of Exception Reporting"(cited 15thOctober 2013), Available from <http://www.qfinance.com/dictionary/exception-reporting>

Sage, (online),"Customer Exception Report Example" (cited 15th October 2013), Available from <http://www.sage.co.uk/sage1000v2_1/form_help/workingw/subfiles/customer_exception_rep_eg.htm>

The National Archives, (online),"Data Protection Act 1988" (cited 14th October 2013), Available from <http://www.legislation.gov.uk/ukpga/1998/29/schedule/1/paragraph/7>

Whiteley, D. (2013) *An Introduction to Information Systems*, Palgrave Macmillan.

APPENDIX A – SQL CODE

```
--------AMANDA MCINTOSH 09081262

DROP TABLE sold_bikes;

DROP TABLE rental_record;

DROP TABLE reservation_record;

--------VAMSI PABBINEEDI 12019750

DROP TABLE maintenance_record;

DROP TABLE  bike_record;

--------PRECIOUS IGBINOSUN 13129589

DROP TABLE manufactLink;

DROP TABLE partsOrder;

DROP TABLE part_record;

DROP TABLE order_record;

DROP TABLE manufacturer_record;

--------VAMSI PABBINEEDI 12019750

DROP TABLE dealer_record;

DROP TABLE class_record;

DROP TABLE size_record;


--------AMANDA MCINTOSH 09081262

DROP TABLE enquire_record;

DROP TABLE customer_record;

DROP TABLE employee_record;

DROP SEQUENCE ren_autono;

DROP SEQUENCE res_autono;

DROP SEQUENCE enq_autono;
```

```
------------------------------CREATE TABLES----------------------------------

--------AMANDA MCINTOSH 09081262

CREATE TABLE employee_record(

employeeNo NUMBER(10) CONSTRAINT employeeNo_pk PRIMARY KEY,

empForename VARCHAR2(20),

empSurname VARCHAR2(20) CONSTRAINT surname_nn NOT NULL,

empAddress VARCHAR2(50) CONSTRAINT address_nn NOT NULL,

empPCode VARCHAR2(8) CONSTRAINT pcode_nn NOT NULL,

empDOB DATE,

empTel VARCHAR2(11),

empStartDate DATE,

empjobRole VARCHAR2(20),

empSal NUMBER (8,2) CONSTRAINT salary_check CHECK (empSal < 1000000),

empEmail VARCHAR2(30),

empUsername VARCHAR2(15),

empPassword VARCHAR2(15),

empEXT VARCHAR2(5),

managerID NUMBER(10) CONSTRAINT fk_managerID REFERENCES employee_record(employeeNo));


CREATE TABLE customer_record(

custNo NUMBER(10) CONSTRAINT custNo_pk PRIMARY KEY,

custForname VARCHAR2(20),

custSurname VARCHAR2(20) CONSTRAINT custForname_nn NOT NULL,

custAddress VARCHAR2(50) CONSTRAINT custAddress_nn NOT NULL,

custPCode VARCHAR2(8) CONSTRAINT custPcode_nn NOT NULL,

custDOB DATE,
```

```
custEmail VARCHAR2(30) CONSTRAINT custEmail_nn NOT NULL CONSTRAINT uk_custEmail UNIQUE,
custTel VARCHAR2(11),
custMob VARCHAR2(11));
CREATE TABLE enquire_record (
enquiryNo NUMBER(10) CONSTRAINT enquiryNo_pk PRIMARY KEY,
enquiryInfo VARCHAR2(100) CONSTRAINT enquiryInfo_nn NOT NULL,
enquiryDate DATE,
note VARCHAR2(100),
custNo NUMBER(10) CONSTRAINT fk_custNo REFERENCES customer_record(custNo),
employeeNo NUMBER(10) CONSTRAINT fk_employeeNo REFERENCES employee_record(employeeNo));


----------CREATE SEQUENCES----------
CREATE SEQUENCE enq_autono
START WITH 1
INCREMENT BY 1
CACHE 100;

CREATE SEQUENCE res_autono
START WITH 1
INCREMENT BY 1
CACHE 100;

CREATE SEQUENCE ren_autono
START WITH 1
INCREMENT BY 1
CACHE 100;
```

--------VAMSI PABBINEEDI 12019750

--Size

```
CREATE TABLE size_record(
sizeID NUMBER(10) CONSTRAINT sizeID_pk PRIMARY KEY,
sizeType VARCHAR2(50) CONSTRAINT sizeType_nn NOT NULL);
```

--Class

```
CREATE TABLE class_record(
classID NUMBER(10) CONSTRAINT classID_pk PRIMARY KEY,
classType VARCHAR2(50) CONSTRAINT classType_nn NOT NULL,
fullDayRate NUMBER(7,2) CONSTRAINT fullDayRate_nn NOT NULL,
halfDayRate NUMBER(7,2) CONSTRAINT halfDayRate_nn NOT NULL  );
```

--Dealer Record

```
CREATE TABLE dealer_record (
dlrID NUMBER (10) CONSTRAINT dlrName_pk PRIMARY KEY,
dlrAddress VARCHAR2(50) CONSTRAINT dlrAddress_nn NOT NULL,
dlrPcode VARCHAR2 (8) CONSTRAINT dlrPcode_nn NOT NULL,
dlrName VARCHAR2(20),
dlrTel VARCHAR2(11),
dlrMob VARCHAR2(11),
dlrEmail VARCHAR2(30) CONSTRAINT Email_nn NOT NULL
CONSTRAINT dlrEmail_uk UNIQUE );
```

```
--------PRECIOUS IGBINOSUN 13129589
--Manufacturer Record
CREATE TABLE manufacturer_Record(
manufactID NUMBER(10) CONSTRAINT manurec_manufactID_pk PRIMARY KEY,
manufactName VARCHAR2(30) CONSTRAINT manurec_manufactName_nn NOT NULL,
manufactAddress VARCHAR2(60) CONSTRAINT manurec_manufactAddress_nn NOT NULL,
manufactPcode VARCHAR2(8) CONSTRAINT manurec_manufactPcode_nn NOT NULL,
manufactTel VARCHAR2(11),
manufactmail VARCHAR2(30) CONSTRAINT manurec_manufactEmail_nn NOT NULL
CONSTRAINT manurec_manufactEmail_uk UNIQUE,
manufactMob VARCHAR2(11),
manufactWeb VARCHAR2(30));
--Order_
CREATE TABLE order_record(
orderNo NUMBER(10) CONSTRAINT order_orderNo_pk PRIMARY KEY,
orderDate DATE CONSTRAINT order_orderDate_nn NOT NULL,
subtotal NUMBER(7,2),
VAT NUMBER(7,2),
Total NUMBER(7,2));
--Part
CREATE TABLE part_record(
partNo NUMBER(10) CONSTRAINT part_partNo_pk PRIMARY KEY,
partname VARCHAR2(30) CONSTRAINT part_partname_nn NOT NULL,
price NUMBER(7,2) CONSTRAINT part_price_nn NOT NULL);
--Parts Order
CREATE TABLE partsOrder(
```

```
Qty NUMBER(4) CONSTRAINT partsOrder_qty_nn NOT NULL,

lineTotal NUMBER(7,2),

orderNo NUMBER(10) CONSTRAINT partsOrder_orderNo_fk REFERENCES order_record(orderNo),

partNo NUMBER(10) CONSTRAINT partsOrder_partNo_fk REFERENCES part_record(partNo),

CONSTRAINT partOrder_comp_pk PRIMARY KEY (orderNo, partNo));

--Manufact Link

CREATE TABLE manufactLink(

manufactID NUMBER(10) CONSTRAINT manufactlink_manufactID_fk REFERENCES
manufacturer_record(manufactID),

partNo NUMBER(10) CONSTRAINT manufactlink_partNo_fk REFERENCES part_record(partNo),

CONSTRAINT manufactlink_comp_pk PRIMARY KEY (manufactID, partNo) );

--------VAMSI PABBINEEDI 12019750

--Bike Record

CREATE TABLE bike_Record(

bikeID NUMBER(10) CONSTRAINT bikeID_pk PRIMARY KEY,

bikeMake VARCHAR2(15) CONSTRAINT bikeMake_nn NOT NULL,

bikeModel VARCHAR2(15) CONSTRAINT bikeModel_nn NOT NULL,

purchaseDate DATE  CONSTRAINT purchaseDate_nn NOT NULL,

purchasePrice NUMBER (7,2) CONSTRAINT purchasePrice_ck CHECK(purchasePrice>0),

writeOffDate DATE,

sizeID NUMBER(10) CONSTRAINT sizeID_fk REFERENCES  size_record(sizeID),

classID NUMBER(10) CONSTRAINT classID_fk REFERENCES  class_record(classID),

manufactID NUMBER(10) CONSTRAINT manufactID_fk REFERENCES  manufacturer_record(manufactID));


CREATE TABLE maintenance_record(

maintenanceID NUMBER(10) CONSTRAINT maintenanceID_pk PRIMARY KEY,

maintenanceDate DATE  CONSTRAINT maintenanceDate_nn NOT NULL,
```

```
faultDesc VARCHAR2(50),
faultDate DATE CONSTRAINT faultDate_nn NOT NULL,
actionTakenDate DATE,
actionTaken VARCHAR2(30) CONSTRAINT actionTaken_nn NOT NULL,
employeeNo NUMBER(10) CONSTRAINT employeeNo_fk REFERENCES  employee_record(employeeNo),
bikeID NUMBER(10) CONSTRAINT bikeID_fk REFERENCES  bike_record(bikeID));
--------AMANDA MCINTOSH 09081262
CREATE TABLE reservation_record(
resNo NUMBER(10) CONSTRAINT resNo_pk PRIMARY KEY,
resDate DATE CONSTRAINT resDate_nn NOT NULL,
custNo NUMBER(10),
CONSTRAINT fk_resCustNo FOREIGN KEY (custNo) REFERENCES customer_record(custNo),
employeeNo NUMBER(10) CONSTRAINT fk_resEmployeeNo REFERENCES employee_record(employeeNo),
bikeID NUMBER(10) CONSTRAINT resBikeID_fk REFERENCES  bike_record(bikeID));

CREATE TABLE rental_record (
rentalNo NUMBER(10) CONSTRAINT rentalNo_pk PRIMARY KEY,
rentPaid NUMBER (7,2) CONSTRAINT rentPaid_ck CHECK(rentPaid > 0),
rentDate DATE  CONSTRAINT renDate_nnNOT NULL,
rentTimeOut VARCHAR2(5) CONSTRAINT renTime_nn NOT NULL,
rentTimeDue VARCHAR2(5),
rentTimeAct VARCHAR2(5),
custNo NUMBER(10) CONSTRAINT fk_renCustNo REFERENCES customer_record(custNo),
bikeID NUMBER(10) CONSTRAINT fk_renBikeID REFERENCES bike_record(bikeID),
employeeNo NUMBER(10) CONSTRAINT fk_renEmpNo REFERENCES employee_record(employeeNo)
);
```

--------AMANDA MCINTOSH 09081262 , VAMSI PABBINEEDI 12019750

```
CREATE TABLE sold_bikes (
bikeID NUMBER(10) CONSTRAINT fk_soldBikeID REFERENCES bike_record(bikeID),
dlrID NUMBER(10) CONSTRAINT fk_soldDlrID REFERENCES dealer_record(dlrID),
CONSTRAINT soldBikecomp_pk PRIMARY KEY (bikeID,dlrID),
saleDate DATE,
salePrice NUMBER (7,2) CONSTRAINT salePrice_ck CHECK(salePrice > 0)
);
```

--------AMANDA MCINTOSH 09081262

----------INSERT DATA----------

--CUSTOMER

```
INSERT INTO customer_record VALUES
(1, 'Amanda', 'Smith', '16 Bury Old Road, Bury', 'BL8 4HS', '16-JAN-80', 'a.smith@gmail.com', '01617974582',
'07852145523');

INSERT INTO customer_record VALUES
(2, 'Ryan', 'Harrop', '101 Holme Avenue, Ramsbottom, Lancashire', 'BM9 2HE', '05-MAR-90', 'rh446@hotmail.co.uk',
'01614523658', '07836521485');

INSERT INTO customer_record VALUES
(3, 'Sam', 'Hall',  '234 The Close, Manchester', 'M45 7YU', '24-DEC-75', 'hall30@aol.com', '01614528745',
'07825634458');

INSERT INTO customer_record VALUES
(4, 'Ben', 'Green', '1 Waterloo Road, Chester', 'CH7 9KL', '08-FEB-91', 'GreenBean99@sky.com', '01263587452',
'07896541236');
```

```
INSERT INTO customer_record VALUES

(5, 'George', 'Roberts', '50 PeterÕs Square, London', 'LI12 3RF', '19-APR-62', 'robgt@yahoo.com', '01617976582',
'07834572293');


INSERT INTO customer_record VALUES

(6, 'Beth', 'Williams', '42 Coniston Drive, Middleton', 'M24 5YT', '15-JUN-84', 'bettie@hotmail.com', '01612345672',
'07825376654');


INSERT INTO customer_record VALUES

(7, 'Stephanie', 'Farnworth', '66 Perth Avenue, Bolton', 'BL9 6YS', '17-AUG-76', 'stephf@googlemail.com',
'01204933655', '07725673351');


INSERT INTO customer_record VALUES

(8, 'Nikki', 'Jones', '145 Pearl Drive, Manchester', 'M44 5SQ', '25-MAR-90', 'nikk1145@hotmail.co.uk', '01204000333',
'07725699936');


INSERT INTO customer_record VALUES

(9, 'Ashley', 'Phillips', '201 Seymour Close, Ramsbottom', 'BL7 5TH', '01-JAN-88', 'ashphill@gmail.com',
'01706592922', '07855566634');


INSERT INTO customer_record VALUES

(10, 'Billy', 'Smith', '2 Booth Street, Bury', 'BL9 6PP', '03-SEP-89', 'billy-s@mmu.co.uk', '01613457234',
'07855467201');


--EMPLOYEE

INSERT INTO employee_record VALUES

(11, 'Ray', 'Owen', 'Arley Avenue, Bury', 'BL9 6WW', '06-JUN-78', '07835412654', '01-JAN-99', 'Manager', '',
'Ray@rr.co.uk', 'Ray', 'RENTALS', '1011', '');
```

INSERT INTO employee_record VALUES

(12, 'Pete', 'Egan', 'Rudwick Drive, Bury', 'BL9 6PO', '11-JUN-89', '07785412654', '01-JAN-99', 'Deputy Manager','50000.00', 'Pete@rr.co.uk', 'Pete', 'rud00', '1010',11);

INSERT INTO employee_record VALUES

(13, 'Sheila', 'Gates', 'Woodhill Road, Bury', 'BL8 8FR', '30-JAN-60', '07898985256', '01-JAN-99', 'PA','20000.00', 'Sheila@rr.co.uk', 'Sheila', 'sswood0', '1354', 12);

INSERT INTO employee_record VALUES

(14, 'Megan', 'Williams', 'Bankhouse Road, Bury', 'BL9 1DD', '15-DEC-95', '01614568524', '01-SEP-12', 'Apprentice', '5000.00', 'temp@rr.co.uk', 'Temp', '1234', '1355',12);

INSERT INTO employee_record VALUES

(15, 'Alph', 'Monro', 'Parkhills Road, Bury', 'BL8 7NC', '07-APR-91', '07894521542', '01-SEP-10', 'Technician','30000.00', 'Alph@rr.co.uk', 'Alph', 'yeet77', '2000',11);

INSERT INTO employee_record VALUES

(16, 'Bert', 'Alba', 'Grange Road, Bury', 'BL0 8IH', '22-MAR-69', '01612588855', '01-JAN-99', 'Technician', '30000.00', 'Bert@rr.co.uk', 'Bert', 'bertA','2000',11);

INSERT INTO employee_record VALUES

(17, 'Paul', 'Coldridge', 'Market Street, Bury', 'BL0 1TH', '14-FEB-59', '07898542153', '01-SEP-09', 'Parts Manager', '40000.00', 'Paul@rr.co.uk', 'Paul', '00work', '2010',11);

INSERT INTO employee_record VALUES

(18, 'Becky', 'Yates', 'Tottington Road, Tottington', 'BL9 8UY', '05-APR-91', '07895552153', '01-SEP-09', 'Administrator', '15000.00', 'Admin@rr.co.uk', 'Admin', 'password','1001', 12);

```
--ENQUIRE

INSERT INTO enquire_record VALUES

(enq_autono.nextval, 'Customer called asking for price list', '12-FEB-13', 'sent via post', 2, 18);


INSERT INTO enquire_record VALUES

(enq_autono.nextval, 'Emailed to ask opening times', '04-MAR-13', 'replied to email', 3, 18);


INSERT INTO enquire_record VALUES

(enq_autono.nextval, 'Letter received asking for information for a school project',  '20-SEP-13', 'Ray responded via post', 9, 13);


INSERT INTO enquire_record VALUES

(enq_autono.nextval, 'Called to cancel reservation', '23-JUN-13',  'reservation removed', 10, 13);


INSERT INTO enquire_record VALUES

(enq_autono.nextval,  'Price list request', '30-JAN-13', 'sent no customer info given', '', 18);


--------VAMSI PABBINEEDI 12019750

INSERT INTO size_record ( sizeid ,sizetype)

VALUES ( 100 , 'large male' );



INSERT INTO size_record ( sizeid ,sizetype)

VALUES ( 101 , 'standard male' );


INSERT INTO size_record ( sizeid ,sizetype)

VALUES ( 102 , 'small male' );
```

```
INSERT INTO size_record ( sizeid ,sizetype)

VALUES ( 103 , 'standard female' );


INSERT INTO size_record ( sizeid ,sizetype)

VALUES ( 104 , 'child' );


--Class record

INSERT INTO class_record(classid,classtype,fulldayrate,halfdayrate)

VALUES(110,'mountain',16.00, 20.00);


INSERT INTO class_record(classid,classtype,fulldayrate,halfdayrate)

VALUES(111, 'road', 13.00 , 10.00);


INSERT INTO class_record(classid,classtype,fulldayrate,halfdayrate)

VALUES( 112 ,'tandem', 33.00 , 27.00 );

--Dealer record

INSERT INTO dealer_record(dlrid,dlrtel,dlrmob,dlremail,dlrpcode,dlraddress,dlrname)

VALUES( 120 ,'01143691476','0736553422','joe.stevens@jsbikez.com','SK17 4JU','15 Back Lane ,Buxton','JS Bikes');


INSERT INTO dealer_record(dlrid,dlrtel,dlrmob,dlremail,dlrpcode,dlraddress,dlrname)

VALUES(121,'01619483479','0775563922','dean.jones@djpeaks.com','M20 3NU','34 Hayfield Road ,Manchester','
Dean Jone Peaks');


INSERT INTO dealer_record(dlrid,dlrtel,dlrmob,dlremail,dlrpcode,dlraddress,dlrname)

VALUES( 122,'01757335892','07700900196','harry.walker@wlkrsride.com','Y026 5KT','145 Bakers Road ,York',
'Walker Rides');
```

INSERT INTO dealer_record(dlrid,dlrtel,dlrmob,dlremail,dlrpcode,dlraddress,dlrname)

VALUES(123,'01914980605','07788604005','deb.yates@dbwheels.com','DH7 5PA','56 Caldwell Street ,Durham','
Debbie Wheels');

INSERT INTO dealer_record(dlrid,dlrtel,dlrmob,dlremail,dlrpcode,dlraddress,dlrname)

VALUES(124,'01484257469','0758143411','kevin.mckann@terraincycle.com','HD5 8VA','48 Ashfield Road
,Hudderfield','Terrain Cycles');

INSERT INTO dealer_record(dlrid,dlrtel,dlrmob,dlremail,dlrpcode,dlraddress,dlrname)

VALUES(125,'01625369222','07798428345','george.nolan@gnrentals.com','SK11 6TY','20 Harrow Road
,Macclesfield','Nolan Bike Rentals');

--------PRECIOUS IGBINOSUN 13129589

--QUERY Manufacuturer_Record

INSERT INTO manufacturer_record

VALUES (1853, 'Techcorporation', '19 Burford Street Hoddeson', 'EN11 8HU', '01617801896',
'info@techcoporation.co.uk', '07852365412', 'Techcorporation.co.uk');

--QUERY Manufacuturer_Record

INSERT INTO manufacturer_record

VALUES (20811, 'Stimholdings', 'Church Street, Claverley, Wolverhampton', 'WV5 7DS', '01617801896',
'info@Stimholdings.co.uk', '0726666444', 'Techcorporation.co.uk');

INSERT INTO manufacturer_record

VALUES (1549, 'sundex', ' B4332, Boncath, Pembrokeshire ', 'SA37 0ER', '03069990437', 'EmmaShah@sun-dex.org ',
'07943335967', 'sun-dex.org');

INSERT INTO manufacturer_record

VALUES (70284, 'Techcan', '38 Lower Breakish, Breakish, Isle of Skye, Highland', 'IV42 8QA', '03169990606', 'RubyKelly@techcan.net', '07800320178', 'techcan.net');

INSERT INTO manufacturer_record

VALUES(9586, 'Kontouch', '3 Green Terrace, Bradford, West Yorkshire', 'BD2 4PH', '03069990888', 'TobyTodd@kon-touch.co.uk ', '07064804971', 'Kon-touch.co.uk');

--------VAMSI PABBINEEDI 12019750

INSERT INTO bike_record (bikeID,bikeMake,bikeModel,purchaseDate,purchasePrice,writeOffDate, sizeID, classID, manufactID)

VALUES(36200,'Superbike','Explorer','30-JAN-13','350.00','15-FEB-14',100 ,110,1853);

INSERT INTO bike_record(bikeID,bikeMake,bikeModel,purchaseDate,purchasePrice,writeOffDate, sizeID, classID, manufactID)

VALUES(36201,'Atom','Speedster','4-JAN-13','400.00','28-FEB-14',101,111,20811);

INSERT INTO bike_record(bikeID,bikeMake,bikeModel,purchaseDate,purchasePrice,writeOffDate, sizeID, classID, manufactID)

VALUES(36202 ,'Razer','Sharp','12-JAN-13','256.00','5-FEB-14',102,112,1549);

INSERT INTO bike_record(bikeID,bikeMake,bikeModel,purchaseDate,purchasePrice,writeOffDate, sizeID, classID, manufactID)

VALUES(36203,'Knight','Excel','25-JAN-13','480.00','10-MAR-14',103 ,110,70284);

INSERT INTO bike_record(bikeID,bikeMake,bikeModel,purchaseDate,purchasePrice,writeOffDate, sizeID, classID, manufactID)

VALUES(36204 ,'Neo','Ritz','18-FEB-13','157.00','31-MAR-14',104 ,112,9586);

INSERT INTO bike_record(bikeID,bikeMake,bikeModel,purchaseDate,purchasePrice,writeOffDate, sizeID, classID, manufactID)

VALUES(36205 ,'Voltech','Spark','15-MAR-13','220.00','31-MAR-14',102,111,1853);

INSERT INTO maintenance_record(maintenanceid,
maintenancedate,faultdesc,faultdate,actiontaken,actiontakendate, bikeid)

VALUES(1,'15-FEB-14','Puncture-rear','15-FEB-14','New Inner Tube','16-FEB-14',36200);

INSERT INTO maintenance_record(maintenanceid, maintenancedate,faultdesc,faultdate,actiontaken
,actiontakendate, bikeid)

VALUES(2,'10-MAR-14','Chain keeps breaking','10-MAR-14','Fit New Chain','27-AUG-14',36201);

INSERT INTO maintenance_record(maintenanceid, maintenancedate,faultdesc,faultdate,actiontaken
,actiontakendate, bikeid)

VALUES(3,'8-FEB-14','Not Applicable','8-FEB-14','NormalService','01-MAR-14', 36202);

INSERT INTO maintenance_record(maintenanceid, maintenancedate,faultdesc,faultdate,actiontaken
,actiontakendate, bikeid)

VALUES(4,'22-MAR-14','Not Applicable','22-MAR-14','Normal Service','3-MAR-14',36203);

INSERT INTO maintenance_record(maintenanceid, maintenancedate,faultdesc,faultdate,actiontaken
,actiontakendate, bikeid)

VALUES(5,'14_FEB-14','Split Saddle','14_FEB-14','Fit new saddle','15-FEB-14',36204);

INSERT INTO maintenance_record(maintenanceid, maintenancedate,faultdesc,faultdate,actiontaken
,actiontakendate, bikeid)

VALUES(6,'17-JAN-14','Faulty brakes','17-JAN-14','New brakes fitted','15-FEB-14',36205);

--------AMANDA MCINTOSH 09081262

INSERT INTO rental_record VALUES (ren_autono.nextval, 33.00, '13-MAR-13', '09:30', '16:30', '16:30', 4, 36204, 13);

INSERT INTO rental_record VALUES (ren_autono.nextval, 27.00, '30-APR-13','09:30', '16:30', '16:30', 1, 36202, 13);

INSERT INTO rental_record VALUES (ren_autono.nextval, 40.00, '01-MAR-13', '12:30', '16:30' , '16:20',5, 36202, 13);

```
INSERT INTO rental_record VALUES (ren_autono.nextval, 50.00, '01-MAR-13', '10:15', '16:30', '16:30', 1, 36203, 14);

INSERT INTO rental_record VALUES (ren_autono.nextval, 80.00, '01-MAR-13', '09:30', '16:30', '18:42', 7, 36202, 11);

INSERT INTO rental_record VALUES (ren_autono.nextval, 100.00, '01-MAR-13', '09:30', '16:30', '18:30', 1, 36202, 14);

INSERT INTO rental_record VALUES (ren_autono.nextval, 40.00, '21-APR-13',  '09:30', '16:00' , '16:30',5, 36205, 13);

INSERT INTO rental_record VALUES (ren_autono.nextval, 60.00, '11-MAR-13',  '11:30', '16:30', '19:30', 1, 36202, 11 );

INSERT INTO rental_record VALUES (ren_autono.nextval, 100.00, '03-MAY-13',  '13:30', '18:30' , '16:30',1, 36203,
11);

INSERT INTO rental_record VALUES (ren_autono.nextval, 30.00, '03-MAY-13',  '13:30', '18:30' , '13:35',4, 36204, 11);


--------PRECIOUS IGBINOSUN 13129589

INSERT INTO order_record

VALUES(489568, '01-Jan-14', 2000,  18.18, 2018.18);


INSERT INTO order_record

VALUES(126850, '15-Feb-14', 823.23, 3.60, 826.23);


INSERT INTO order_record

VALUES(25689, '19-Jan-14', 48.23, 0.58, 48.81);


INSERT INTO order_record

VALUES(156895, '13-Jan-14', 320, 0.23, 320.23);


INSERT INTO order_record

VALUES(295894, '16-Feb-14', 589.01, 0.23, 589.01);


INSERT INTO order_record

VALUES(568974, '15-Jan-14', 1235, 0.95, 1235.95);
```

```
INSERT INTO order_record
VALUES(132568, '13-Feb-14', 245, 0.58, 245.58);

INSERT INTO order_record
VALUES(89456, '07-Feb-14', 589, 0.99,  589.99);

INSERT INTO order_record
VALUES(956426, '14-Jan-14', 582, 0.85, 582.85);

INSERT INTO order_record
VALUES(456789, '03-Jan-14', 235, 0.25, 235.25);

INSERT INTO part_record
VALUES (1005, 'Rusty Alarm', 22.02);
INSERT INTO part_record
VALUES (426, 'Moving Scissors', 45.25);

INSERT INTO part_record
VALUES (456, 'Ivory Brown Dinosaur', 85.98);

INSERT INTO part_record
VALUES (199, 'Long Boomerang', 99.56);

INSERT INTO part_record
VALUES (985, 'long iron', 45.02);
```

```
INSERT INTO part_record
VALUES (178, 'Smart lunar front BikeLight' , 35.49);

INSERT INTO part_record
VALUES (756, 'PanoBike Cadence Speed Sensor' , 38.92);

INSERT INTO manufactLink
VALUES (9586, 1005);
INSERT INTO manufactLink
VALUES (1853, 426);
INSERT INTO manufactLink
VALUES (20811, 199);
INSERT INTO manufactLink
VALUES (1549, 985);
INSERT INTO manufactLink
VALUES (1549, 199);
INSERT INTO manufactLink
VALUES (70284, 178);
INSERT INTO manufactLink
VALUES (9586, 178);
INSERT INTO manufactLink
VALUES (20811, 1005);
INSERT INTO partsOrder
VALUES(5, 110.25, 156895, 1005);
INSERT INTO partsOrder
VALUES(6, 271.50,  295894, 426 );
```

```
INSERT INTO partsOrder

VALUES(2, 171.96,  126850, 456 );

INSERT INTO partsOrder

VALUES(1, 99.56, 456789, 199);

INSERT INTO partsOrder

VALUES(10, 450.20, 489568, 985);

INSERT INTO partsOrder

VALUES(3, 106.47,  132568, 178);

INSERT INTO partsOrder

VALUES(14, 544.88, 89456, 756);

INSERT INTO partsOrder

VALUES(6, 271.50,  956426, 426 );

INSERT INTO partsOrder

VALUES(2, 44.04, 25689, 1005);

INSERT INTO partsOrder

VALUES(14, 544.88, 568974, 756);

INSERT INTO partsOrder

VALUES(1, 99.56, 489568, 199);

INSERT INTO partsOrder

VALUES(10, 450.20, 132568, 985);

INSERT INTO partsOrder

VALUES(3, 106.47,  489568, 178);

INSERT INTO partsOrder

VALUES(14, 544.88,956426, 756);

INSERT INTO partsOrder

VALUES(6, 271.50, 89456, 426 );
```

```
INSERT INTO sold_bikes

VALUES(36205, 120, '15-Dec-14', 200);

INSERT INTO sold_bikes

VALUES(36200, 121, '30-Jun-14', 315.49);


INSERT INTO sold_bikes

VALUES(36201, 122, '28-Sep-14', 358.99);

INSERT INTO sold_bikes

VALUES(36202, 123, '5-Oct-14', 213.58 );

INSERT INTO sold_bikes

VALUES(36203, 124, '10-Nov-14', 413.12 );

INSERT INTO sold_bikes

VALUES(36204, 125, '31-Jul-14', 88.90 );

--------AMANDA MCINTOSH 09081262

INSERT INTO reservation_record VALUES (res_autono.nextval, '13-MAR-14', 1, 13, '36200');

INSERT INTO reservation_record VALUES (res_autono.nextval, '30-APR-14', 1, 13, '36201');

INSERT INTO reservation_record VALUES (res_autono.nextval, '01-MAR-14', 2, 13, '36202');

INSERT INTO reservation_record VALUES (res_autono.nextval, '01-MAR-14', 3, 13, '36203');

INSERT INTO reservation_record VALUES (res_autono.nextval, '01-MAR-14', 4, 13, '36204');

INSERT INTO reservation_record VALUES (res_autono.nextval, '01-MAR-14', 5, 13, '36204');

INSERT INTO reservation_record VALUES (res_autono.nextval, '21-APR-14', 5, 14, '36204');

INSERT INTO reservation_record VALUES (res_autono.nextval, '11-MAR-14', 5, 14, '36201');

INSERT INTO reservation_record VALUES (res_autono.nextval, '03-MAY-14', 6, 14, '36202');

INSERT INTO reservation_record VALUES (res_autono.nextval, '03-MAY-14', 7, 11, '36202');
```

----------QUERIES----------

--------VAMSI PABBINEEDI 12019750

```
SELECT maintenanceid, maintenancedate,faultdesc,faultdate,actiontaken,actiontakendate
FROM maintenance_record;


SELECT faultdesc|| ' '||maintenanceid AS "Bike Fault Type"
FROM maintenance_record;


DESCRIBE maintenance_record


SELECT maintenanceid, maintenancedate,faultdesc,faultdate,actiontaken,actiontakendate
FROM maintenance_record
WHERE maintenanceid = 4;


SELECT maintenanceid,faultdesc
FROM maintenance_record
WHERE faultdate = '15-FEB-14';


SELECT maintenanceid,maintenancedate,faultdate,actiontaken,actiontakendate
FROM maintenance_record
WHERE faultdesc = 'Chain keeps breaking';


SELECT maintenanceid,maintenancedate,faultdesc ,faultdate,actiontaken,actiontakendate
FROM maintenance_record
WHERE actiontakendate < '3-MAR-14';
```

```
SELECT maintenanceid, maintenancedate,faultdesc,faultdate,actiontaken,actiontakendate

FROM maintenance_record

WHERE maintenanceid  BETWEEN 2 AND 5 ;


SELECT maintenanceid, maintenancedate,faultdesc,faultdate,actiontaken ,bikeID ,bikeMake,bikeModel

FROM maintenance_record

JOIN bike_record

USING (bikeID);


SELECT maintenanceid, maintenancedate,faultdesc,faultdate,actiontaken,actiontakendate

FROM maintenance_record

WHERE LOWER(actiontaken) LIKE 'normalservice';


SELECT maintenanceid, maintenancedate,faultdesc,faultdate,actiontaken,actiontakendate

FROM maintenance_record

ORDER BY actiontakendate;


SELECT maintenanceid, maintenancedate,faultdesc,faultdate,actiontaken ,bikeID ,bikeMake,bikeModel
,sizeid,sizetype,classid,classtype

FROM maintenance_record

NATURAL JOIN bike_record

NATURAL JOIN size_record

NATURAL JOIN class_record;


SELECT faultdate, (SYSDATE-actiontakendate)/7 AS WEEKS

 FROM maintenance_record

WHERE maintenanceid = 2;
```

```
FROM maintenance_record
NATURAL JOIN bike_record
NATURAL JOIN size_record
NATURAL JOIN class_record
ORDER BY actiontakendate ;

--------AMANDA MCINTOSH 09081262
--QUERY TO LIST RENT PAID BY MONTH
SELECT to_char(rentDate,'MONTH YYYY')"Month", SUM(rentPaid)"Rent Paid"
FROM rental_record
GROUP BY to_char(rentDate,'MONTH YYYY');

--QUERY TO LIST TOTAL ORDERS PAID BY MONTH
SELECT to_char(orderDate,'MONTH YYYY')"Month", SUM(total)"Outgoings"
FROM order_record
GROUP BY to_char(orderDate,'MONTH YYYY');

--QUERY TO CALCULATE MONTHLY SALARY OUTGOINGS
SELECT ROUND((SUM(empSal)/12),2) "January Sal", ROUND((SUM(empSal)/12),2) "Februry Sal",
ROUND((SUM(empSal)/12),2) "March Sal", ROUND((SUM(empSal)/12),2) "April Sal",
ROUND((SUM(empSal)/12),2) "May Sal",
 ROUND((SUM(empSal)/12),2) "June Sal"
FROM employee_record;

--total the yearly salary , total rent paid as total income and total sold biked
SELECT SUM(empSal) "Yearly Salary", SUM(rentPaid) "Total Income", SUM(salePrice)"Total Sold Bikes"
```

```
FROM employee_record

NATURAL JOIN  rental_record, sold_bikes;


--list total rent paid before today's date

SELECT SUM(rentPaid) "Income"

FROM rental_record

WHERE rentDate < sysdate;


-- select all of the sold bike records

SELECT *

FROM sold_bikes;


--select all of the order records and order by date

SELECT *

FROM order_record

ORDER BY orderDate ASC;


--describe the rental records table structure

describe rental_record


--display the employees details and their salary if the earn more than 10,000 and have been employed before 01-JAN-2010

SELECT employeeNo, empForename|| ' '|| empSurname AS "Employee Name", empSal "Salary", empStartDate "Start Date"

FROM employee_record

WHERE empSal > 10000

AND empStartDate < '01-JAN-10';
```

--------PRECIOUS IGBINOSUN 13129589

-------Describe Statement--------------------------------

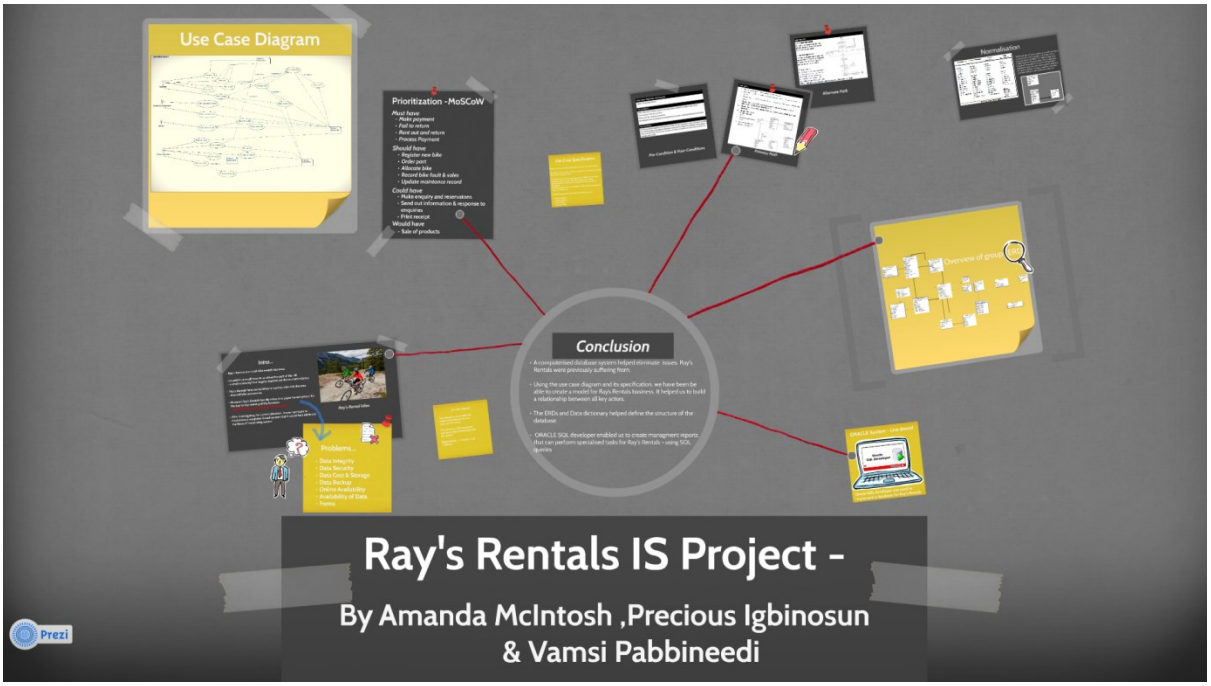
```
DESCRIBE sold_bikes;

DESCRIBE rental_record;

DESCRIBE reservation_record;

DESCRIBE maintenance_record;

DESCRIBE  bike_record;

DESCRIBE manufactLink;

DESCRIBE partsOrder;

DESCRIBE part_record;

DESCRIBE order_record;

DESCRIBE manufacturer_record;

DESCRIBE dealer_record;

DESCRIBE class_record;

DESCRIBE size_record;

DESCRIBE enquire_record;

DESCRIBE customer_record;

DESCRIBE employee_record;
```

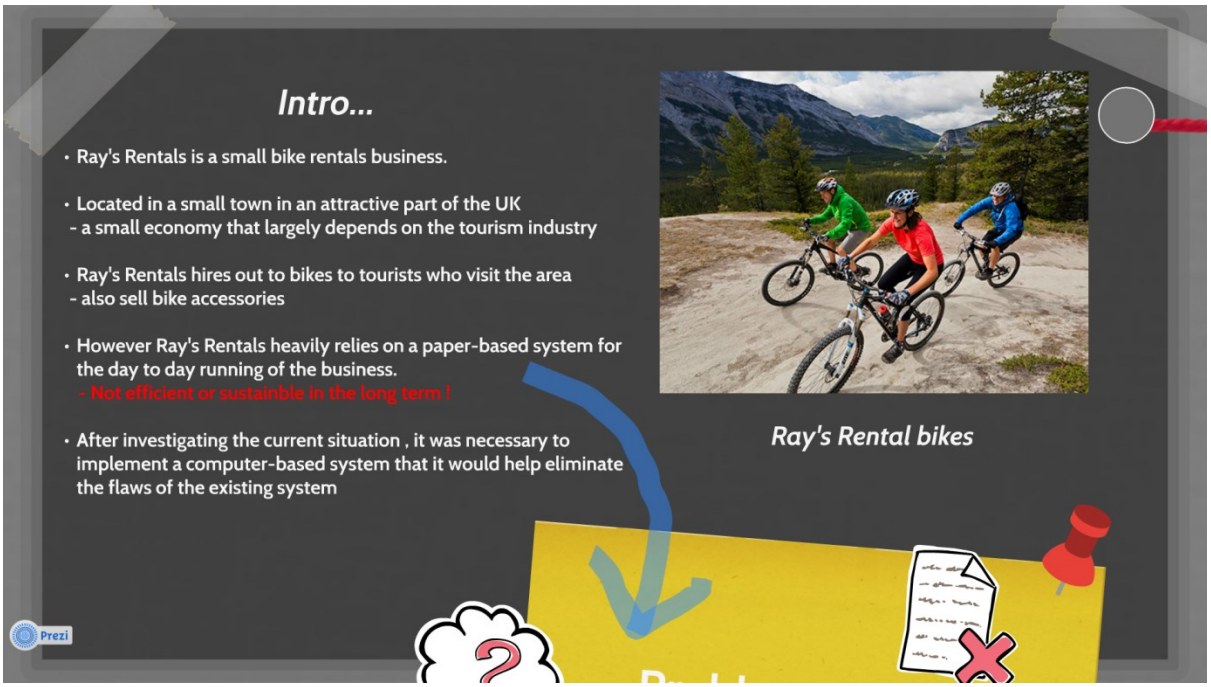
----------SELECT STATEMENTS------------------

```
SELECT bikeid, sizeid, sizetype

FROM size_record

NATURAL JOIN bike_record;

SELECT  bikeid, manufactid, manufactname,  bikemodel, sizeid, sizetype, dlrid, dlrname, maintenanceid,

ROUND(MONTHS_BETWEEN(SYSDATE,purchasedate),0) AS AGE_OF_BIKE_IN_MONTHS

FROM bike_record
```

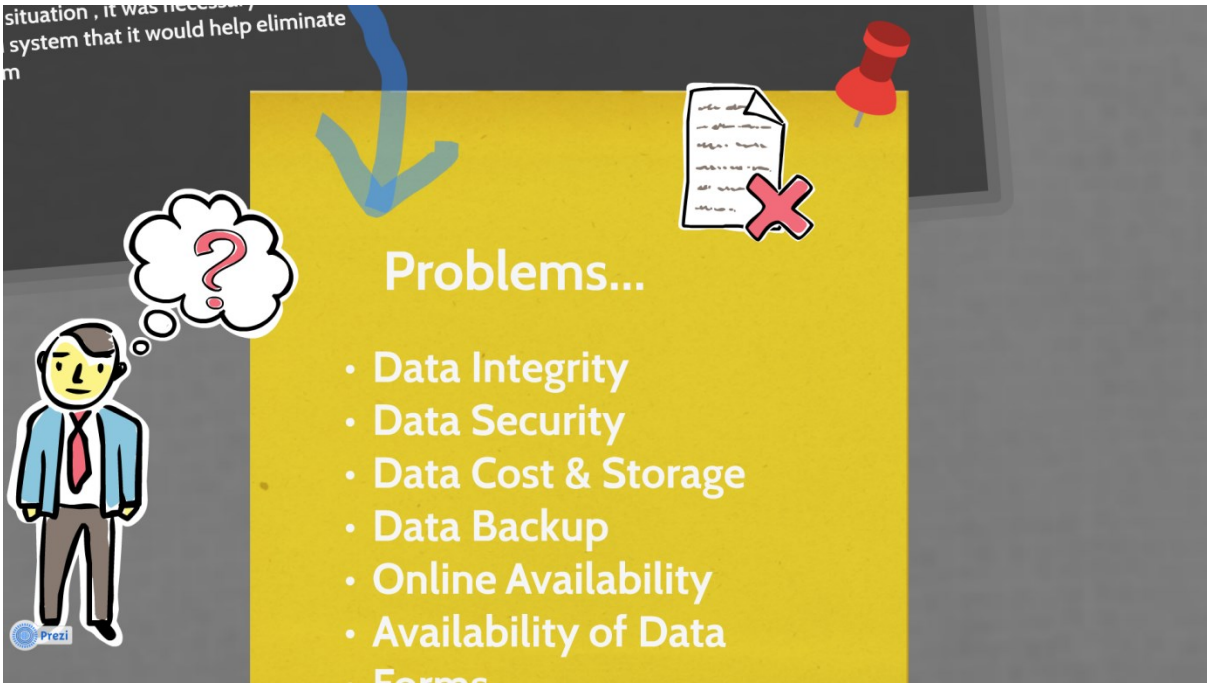
```
NATURAL JOIN size_record

NATURAL JOIN dealer_record

NATURAL JOIN manufacturer_record

NATURAL JOIN maintenance_record

WHERE MONTHS_BETWEEN (SYSDATE, purchasedate) >=12;


SELECT orderno, MIN(total), MAX(total),  AVG(total)

FROM order_record

GROUP BY orderno;


SELECT orderno, orderdate, partno, 3*price AS extra_order, ((3*price) + total) AS new_total

FROM partsorder

NATURAL JOIN order_record

NATURAL JOIN part_record

WHERE orderdate LIKE '%JAN%'

ORDER BY orderno;
```

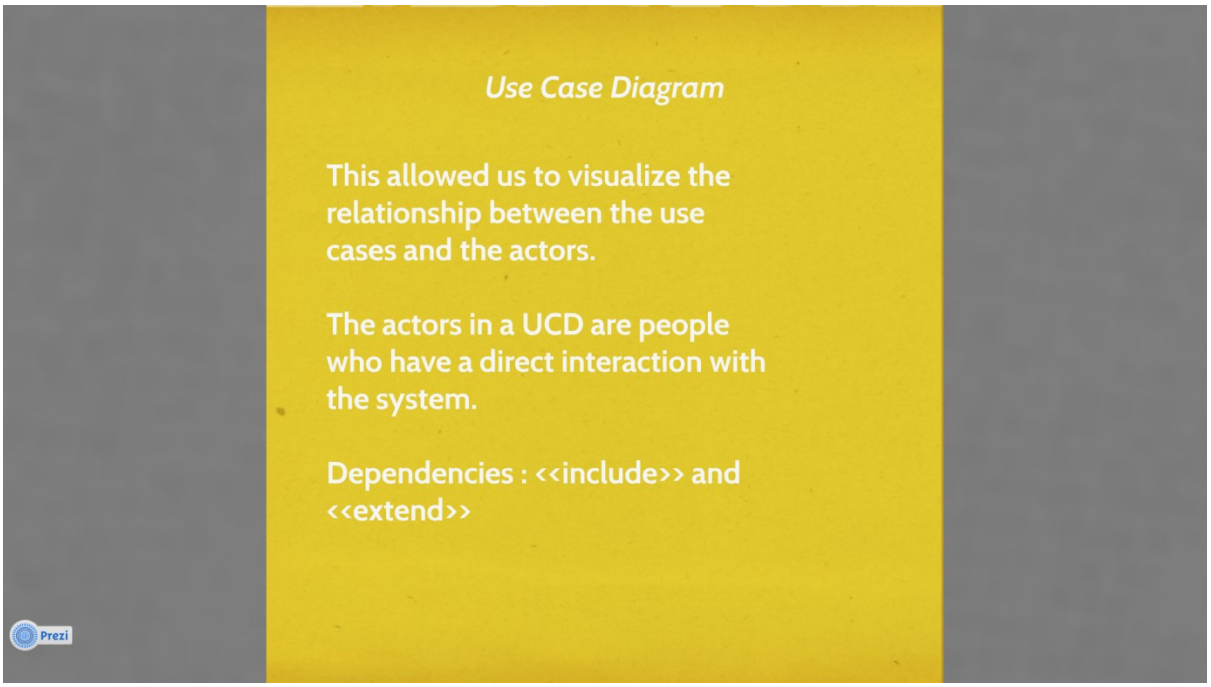
APPENDIX A – PRESENTATION SLIDES



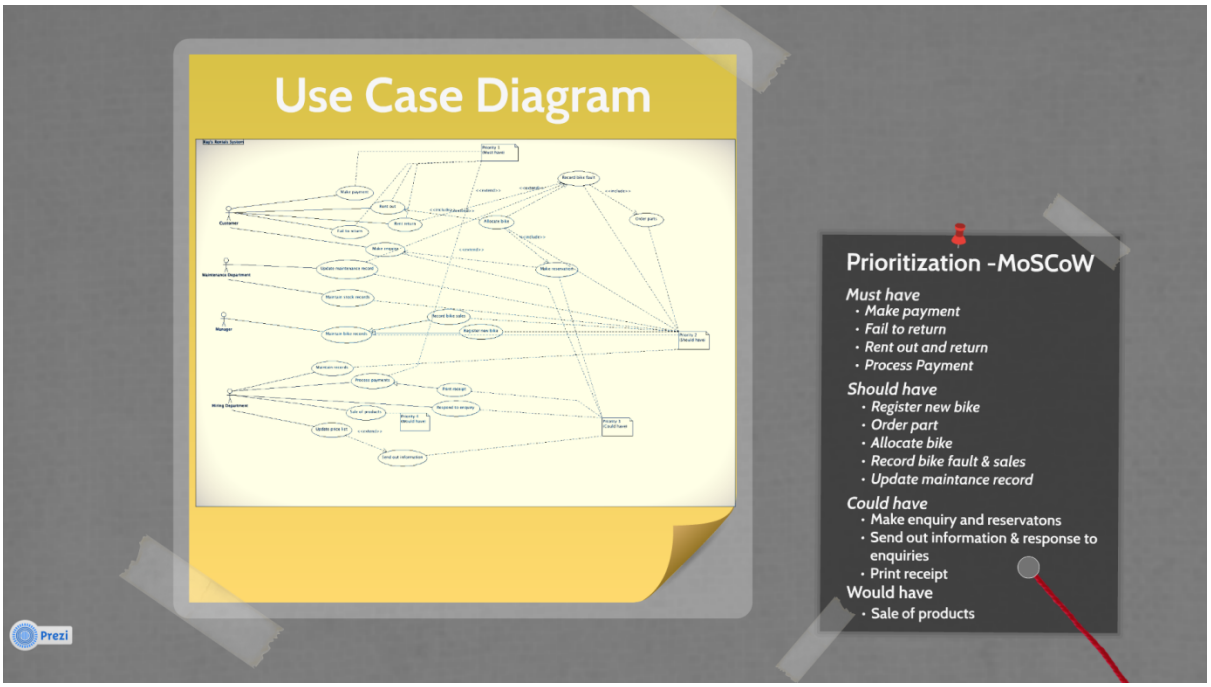


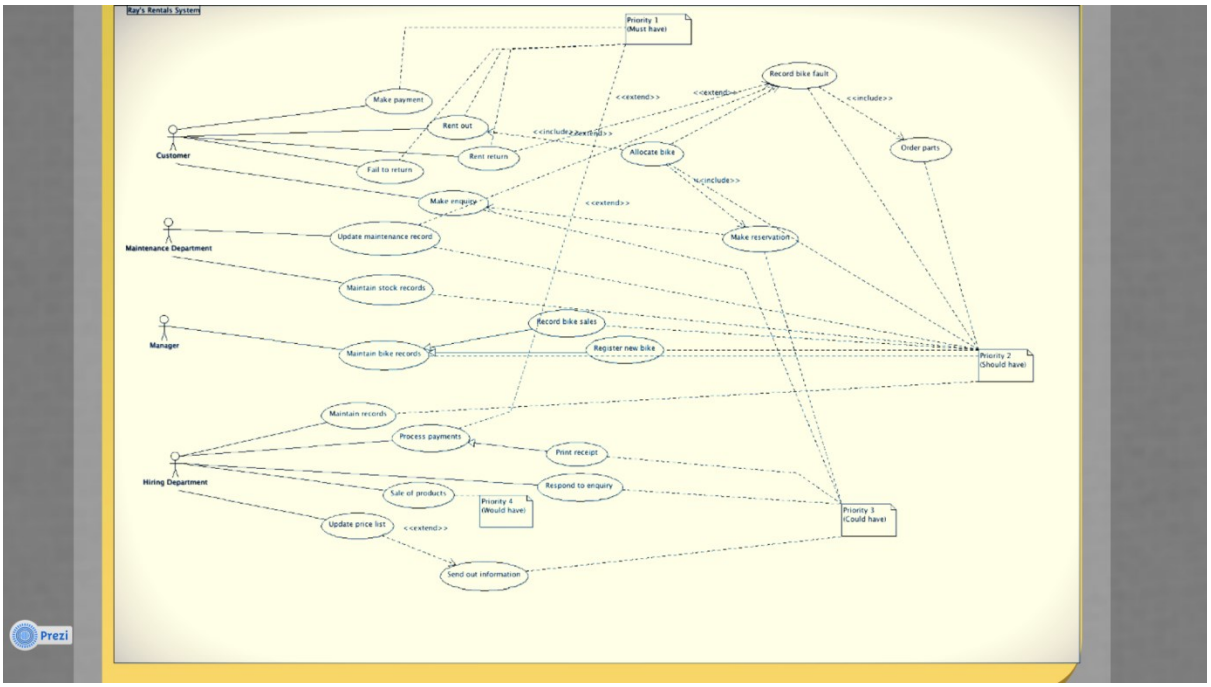
Information Systems Project



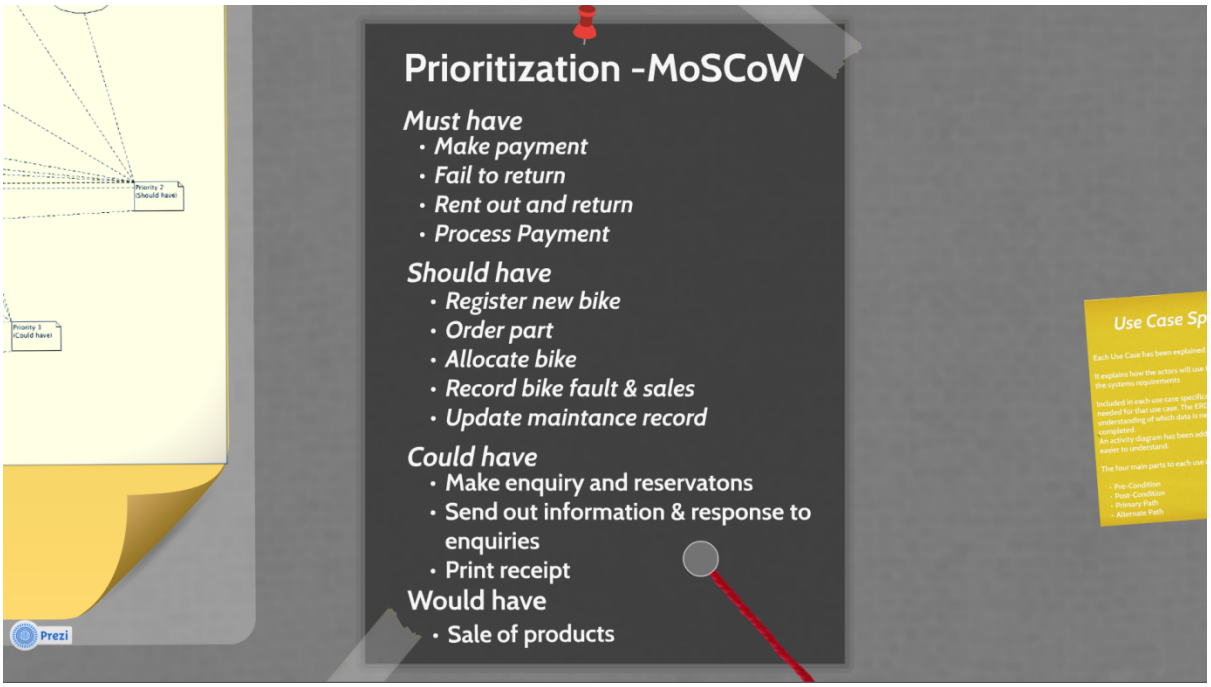


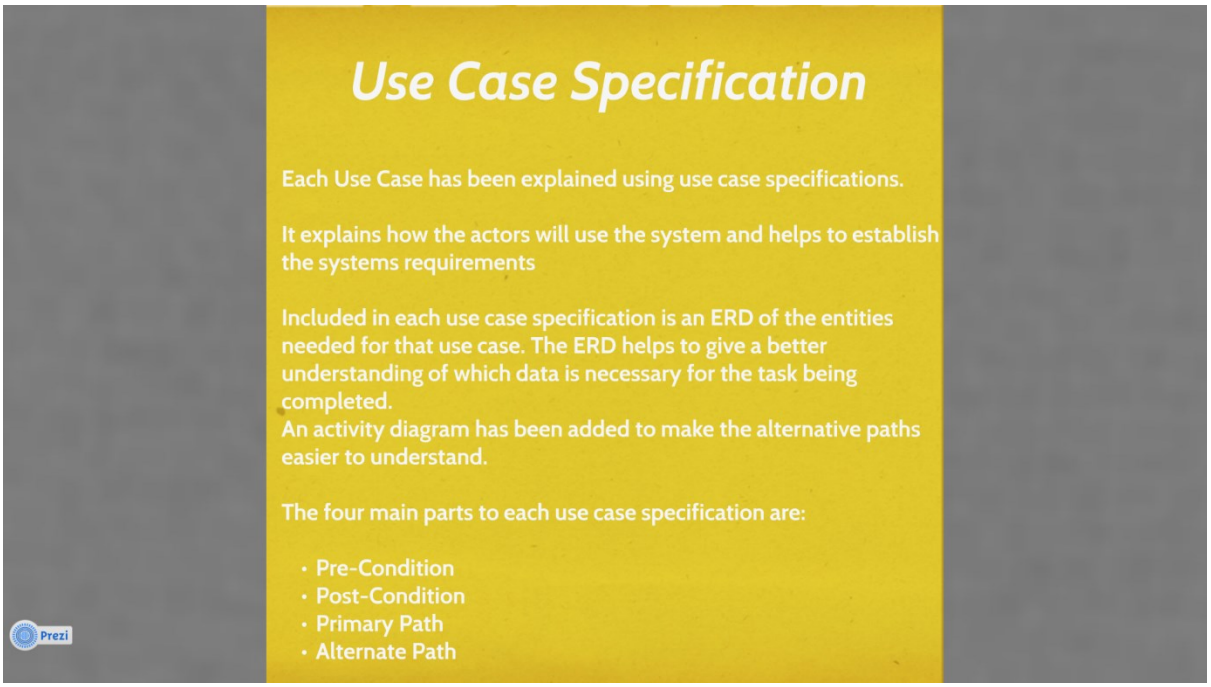
Information Systems Project



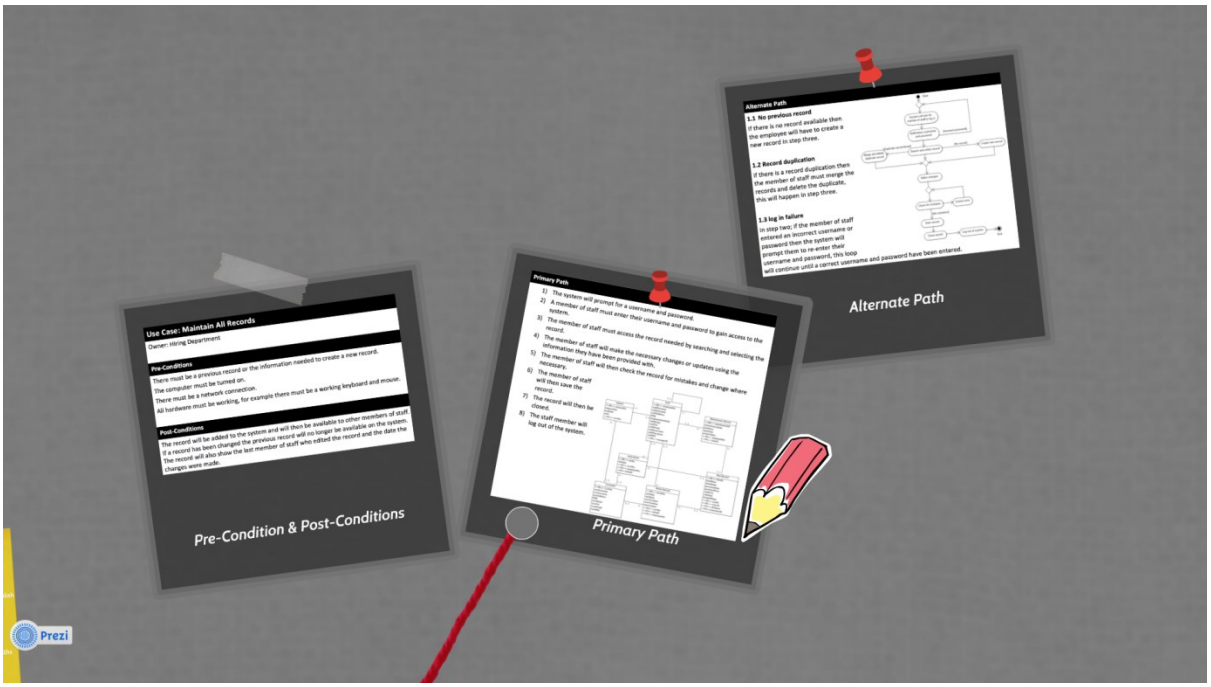


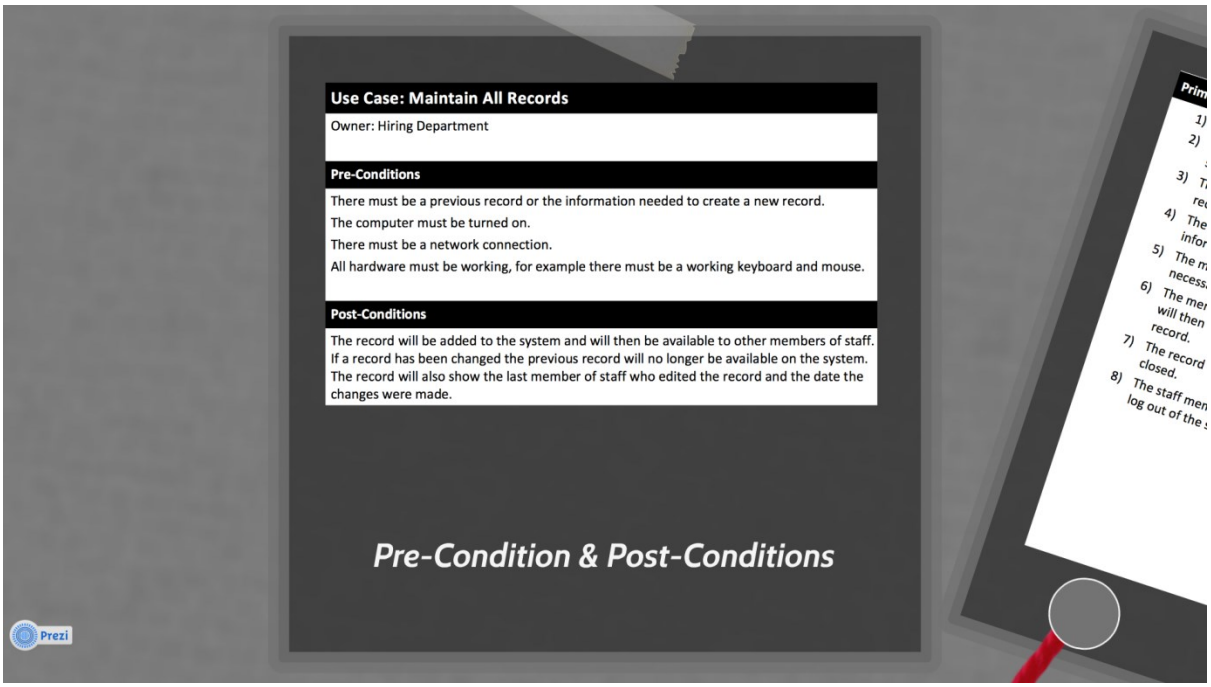
Information Systems Project



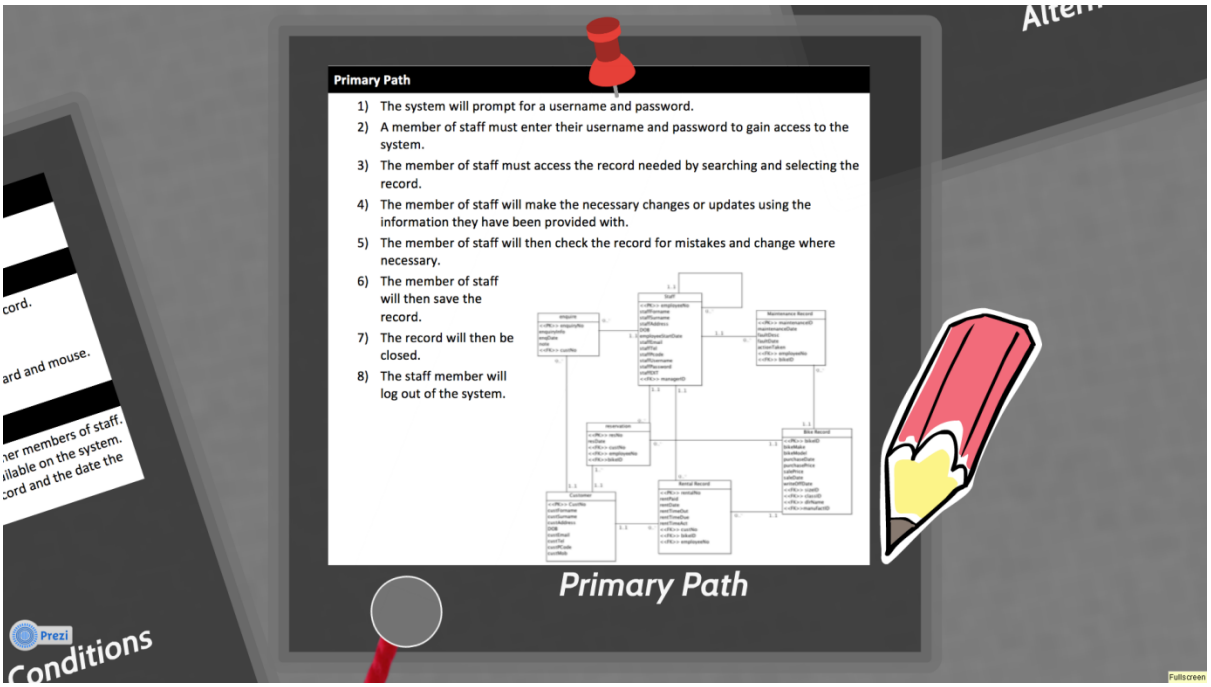


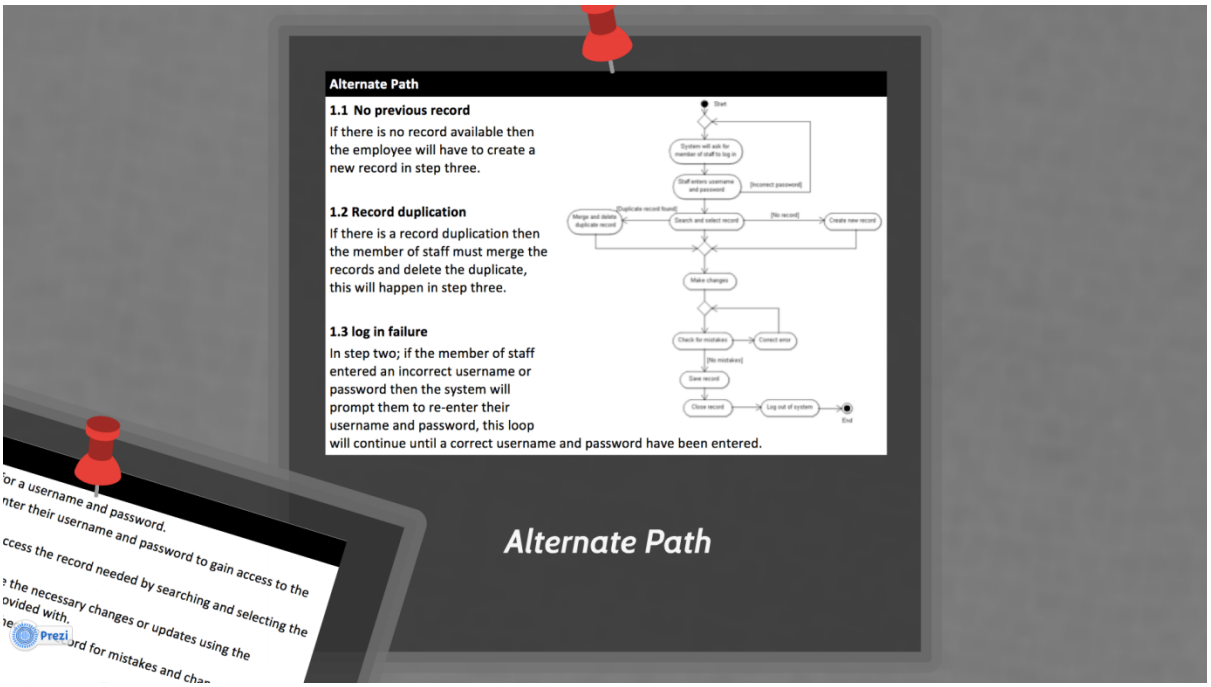
Information Systems Project





Information Systems Project





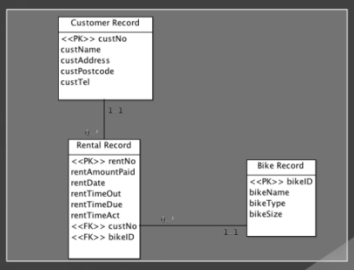
Information Systems Project

Normalisation

Un-Normalised	First Normalisation	Second Normalisation	Third Normalisation
bikeNo	bikeNo	bikeNo	bikeNo
bikeName	bikeName	bikeName	bikeName
bikeType	bikeType	bikeType	bikeType
bikeSize	bikeSize	bikeSize	bikeSize
rentNo			
rentDate	bikeNo*	bikeNo*	bikeNo*
rentTimeOut	rentNo	rentNo	rentNo
rentTimeDue	rentDate	rentDate	rentDate
rentTimeAct	rentTimeOut	rentTimeOut	rentTimeOut
custNo	rentTimeDue	rentTimeDue	rentTimeDue
custName	rentTimeAct	rentTimeAct	rentTimeAct
custAddress	custNo	custNo	custNo*
custPostcode	custName	custName	rentAmountPaid
custTel	custAddress	custAddress	
rentAmountPaid	custPostcode	custPostcode	custNo
	custTel	custTel	custName
	rentAmountPaid	rentAmountPaid	custAddress
			custPostcode
			custTel

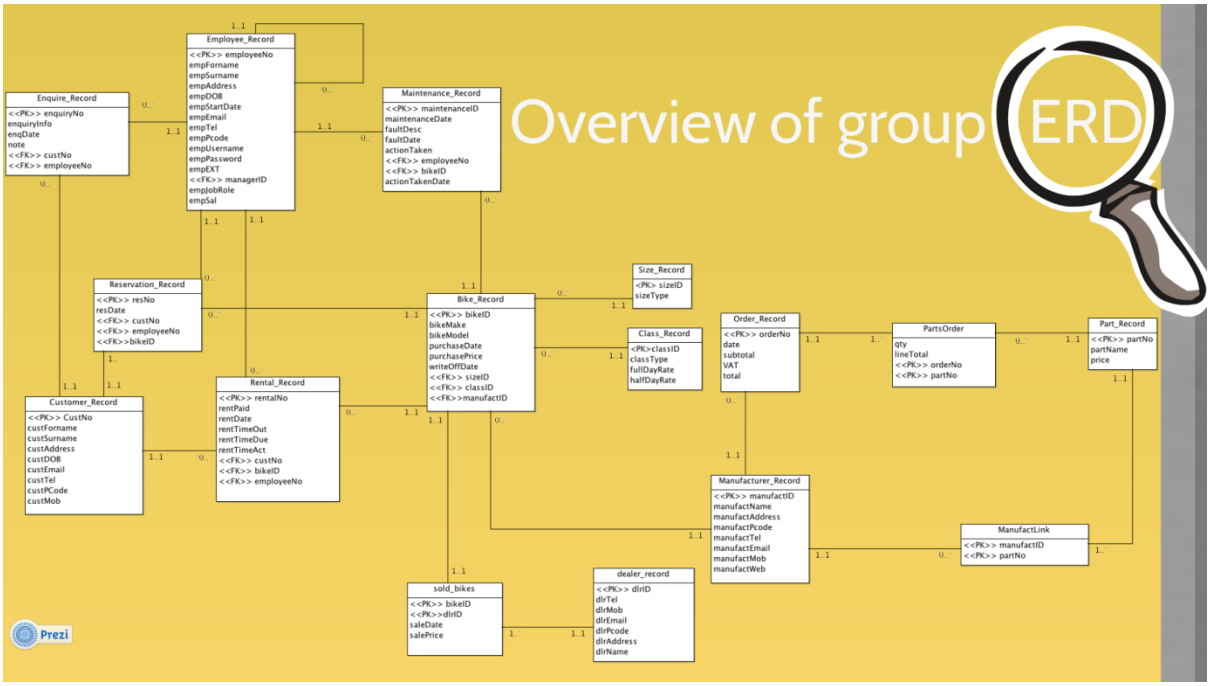
Figure 3.0 RDA for Rental Record

The Rental Record only underlines a small quantity of attributes. After normalisation only three entities were required in this ERD. The relationships were added based on the assumption that a customer's record can be saved on the system without them needing to rent a bike and a bike could be added to the system without it being rented, for example a new bike will have no rental records attached.



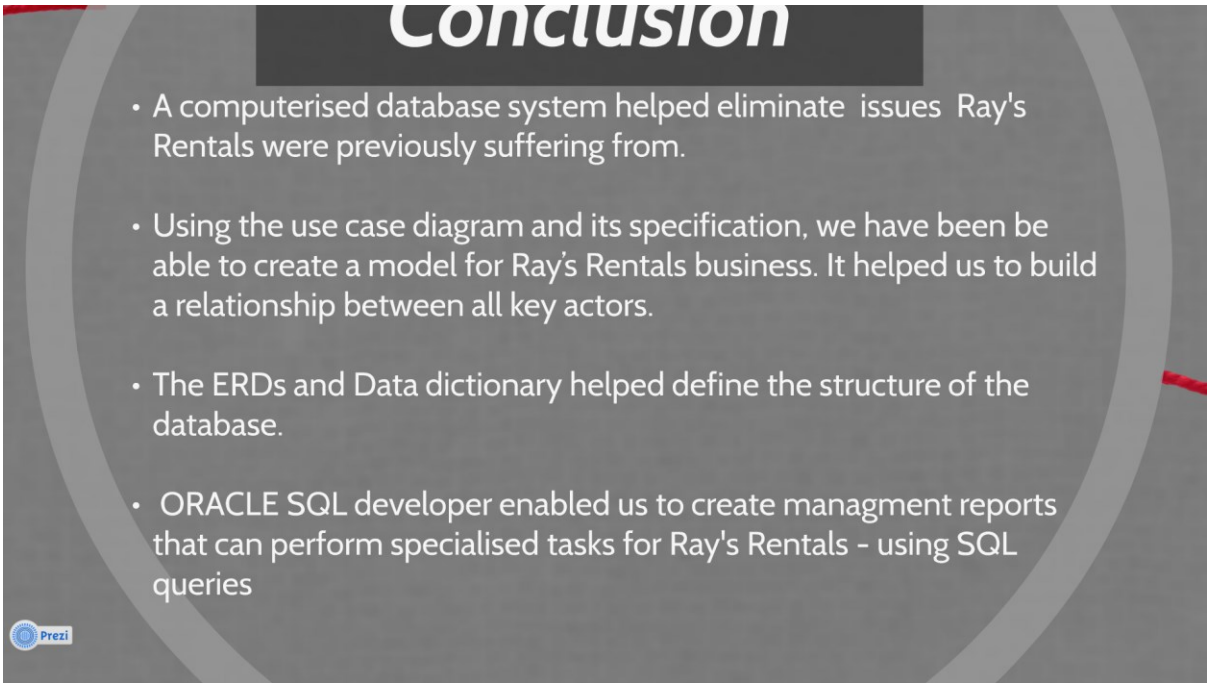
Prezi

Overview of group ERD



Prezi


ORACLE System - Live demo!
Oracle
SQL Developer
ORACLE
Oracle SQL developer was used to implement a database for Ray's Rentals
Prezi


Conclusion
• A computerised database system helped eliminate issues Ray's Rentals were previously suffering from.
• Using the use case diagram and its specification, we have been be able to create a model for Ray's Rentals business. It helped us to build a relationship between all key actors.
• The ERDs and Data dictionary helped define the structure of the database.
• ORACLE SQL developer enabled us to create managment reports that can perform specialised tasks for Ray's Rentals - using SQL queries
Prezi