

Advanced & Distributed Databases - Assessment Criteria and Feedback Sheet

Name: C.Gillman

Mark: 55

Higher levels of achievement are described on the RHS of the grid. Each level subsumes the previous level. Each of the three criteria below contributes a third to the overall mark.

Primarily, Analysis & Design and Learning Outcome CS2				
Does not reach required threshold.	Analysis and design artefacts meet most scenario requirements, but with some deficiencies.	Described analysis and design artefact meet scenario requirements and using appropriate tools and methods.	Rationale for design decisions explained (inc. most higher-level requirements).	Comprehensive, fully justified and documented analysis and design artefacts meet all requirements in full.
Primarily, Implementation and Learning Outcome P&PS4				
Does not reach required threshold.	Implementation artefacts meet most scenario requirements, but with some deficiencies.	Documented implementation and test artefacts meet scenario requirements.	Discussion of encountered implementation issues and solutions (inc. most higher-level requirements).	Comprehensive and fully documented implementation and test artefacts meet all requirements in full.
Primarily, Evaluation Report and Learning Outcomes CS2, K&U1 and P&PS3				
Does not reach required threshold.	Evaluation report meets most requirements but with some deficiencies. Evidence that relevant background reading has been undertaken.	Richer and referenced evaluation against some appropriate criteria. Some logical recommendations developed.	Well written evaluation report against full set of criteria and full set of recommendations developed. Most higher-level requirements addressed. References clearly underpin the evaluation throughout.	Comprehensive and critical evaluation report addressing all requirements, and based on extensive research.

Notes :

1. Normally, artefacts for all of the 5 themes must be submitted and reach the required threshold.
2. Each of the activities indicates work which could contribute to higher-level achievement.

Additional Comments (particularly for borderline and exception cases)

No accounts design.



Southampton Solent University

Advanced and Distributed Database's BSc Computing

1. Exploiting DBMS Data Models and Server Functionality
2. Accessing and Manipulating Data in Applications
3. Improving Data access by Data Distribution and Replication
4. Multi Dimensional Modelling and Analysis for Decision Support
5. Mining Databases for Decision Support
6. X Query within Oracle (Advanced Topic)

Author: Chris Gillman
Topic: Advanced and Distributed Database's
Date: 1st April 2010

Advanced and Distributed Databases

Contents Page

Topic	Page Number
Exploiting DBMS Data Models and Server Functionality:	
1. Introduction	4
2. Background	4
2.1 Triggers	
2.2 XML	
2.3 X Path	
3. Design	6
3.1 Table Design	6
3.2 Sequence Design	
3.3 Database Trigger	
4. Implementation	8
4.1 Insert Record	
4.2 Delete Record	
4.3 Update Record	
4.4 Selecting the XML	
5. Testing	9
6. Conclusion	10
6.1 Recommendation	
7. Advanced Topic	10
8. References	10
Accessing and Manipulating Data in Applications:	
1. Introduction	11
2. Background	11
2.1 Java JDBC/JDO	
2.2 PL/SQL	
2.3 APEX	
3. Design & Implementation	13
4. Testing	14
5. Conclusion	15
5.1 Java JDBC	
5.2 PL/SQL	
5.3 Recommendation	
6. References	16
Improving Data Access by Data Distribution and Replication	
1. Introduction	17
2. Background	17
2.1 Data Replication	
2.2 Key Advantages of Replication	
2.3 Data Distribution	
2.4 Database Links	
2.5 Distributed Queries	

- 3. Design & Implementation 19
 - 3.1 Virtual Table – View
 - 3.2 Snapshot Log
- 4. Distributed Database Design Model
 - 4.1 Explanation behind Model
- 5. Testing 22
- 6. Conclusion 23
- 7. References 23

Multi Dimensional Modelling and Analysis for Decision Support

- 1. Introduction 24
- 2. Background 24
 - 2.1 Different OLAP Tools
- 3. Design & Implementation 26
- 4. Dimension Modelling 27
 - 4.1 Dimension Model
- 5 Testing 31
- 6. Conclusion 33
- 7. References 33

Mining Databases for Decision Support

- 1. Introduction 34
- 2. Background 34
 - 2.1 Market Basket Analysis
 - 2.2 Sample
- 3. Design 35
- 4. Implementation 36
- 5. Testing 37
- 6. Tools & Technologies
 - 6.1 Association Rules
 - 6.2 Product Hierarchies
 - 6.3 Confidence Level
 - 6.4 Drill Down Analysis
- 7. Conclusion 38
- 8. References 39

X Query within Oracle - Advanced Topic

- 1. Introduction 40
- 2. Background 40
 - 2.1 X Query within Oracle
 - 2.2 Examples of its use
- 3. Scenario for the advanced topic 41
- 4. Design 41
- 5. Testing 41
- 6. Conclusion
- 6. References 42

Advanced and Distributed Databases

DBMS Data Models and Server Functionality

1. Introduction

There is a need to further develop product and price information and its availability for the OE database.

The next section of this report will focus on and discuss three different tools which are involved with in database management system's to enhance the server functionality. It is important to have a server with high functionality so that the database user can perform lots of different tasks / actions on the database to get useful results out of the database. The three tools are Triggers, XML, X Path.

2. Background

2.1 Triggers

Triggers are used in databases as a way of auditing and checking for consistency in the database. They can be very useful as they can pick up on changes made to database tables for database managers to check and amend if inaccurate.

A Trigger is code within the database, which executes when the conditions are met. Whatever the outcome of the trigger the final stage will insert a row in to another table. This other table could be an audit table, which will be for the trigger to insert its audit data in to.

There are different types of triggers available with database management systems. Firstly there can be 'Insert Triggers'. These type of triggers are run every time a new row is inserted in to a database table, the trigger will run and place an audit row in to the audit table. An example could be an online shop when a product is inserted in to the database with a price a trigger could be run to insert a record of this entry in to a separate audit table.

Another type of trigger could be an update trigger. This can be a really important type of trigger as it records changes to existing rows in the database. This means that when an update occurs in a database table the trigger will run and record the update as a new row in the audit table. An example of this could be a company with a payroll system where they would like to audit changes to the salary field of the Payroll table to check for fraudulent updates to the database table.

The final type of trigger is very similar as the other two in terms of how they work. This is the delete trigger. When a row is deleted the trigger runs and records what is deleted in the audit table. This can be very important for a company to audit and manage changes to their database.

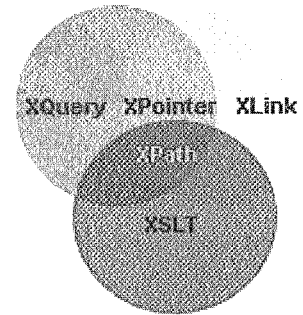
2.2 XML

This tool is a fairly new tool within databases and it is becoming more popular over time as a result of other companies wishing to use data from other companies to publish information. An example of this could be rail information. National Rail Enquires publish all the train times on their database and all the train operators like to show that Information on their sites. This can be done using XML.

It works because database management systems have a built-in function called 'XML Type'. Within this field in a table the contents of that particular field can be inserted as XML. As a result of this other companies can read this XML and develop their sites to show this Information. The main company can then make changes to their tables and update their XML.

2.3 X Path

This is used to navigate through XML documents and elements within the document. It is the syntax for defining the parts of an XML document. It uses path expressions to navigate in XML documents. The path expressions are very similar to computer paths and filenames. They are used to select nodes or node sets in of an XML document. There are hundreds of built in functions to increase the server functionality. These can include, string values, numeric values, date and time comparison and Boolean values. The diagram on the right shows how X path is the framework for the other XML technologies.



References ?

3. Design

3.1 Table Design

```
//Chris Gillman - Databases Assignment 2009
//Southampton Solent University
//Insert Tables in to the DBMS
```

```
CREATE TABLE PRODUCT_INFORMATION
(
Item_ID int NOT NULL PRIMARY KEY,
Category varchar(255),
Weight_Class varchar(255),
Warranty_Class varchar(255),
Warrenty varchar(255),
Supplier_ID int,
Product_Status varchar(255),
List_Price NUMBER(10,2),
Min_Price NUMBER(10,2),
Catalog_URL varchar(255),
FOREIGN KEY (Item_ID)
REFERENCES PRODUCT_DESCRIPTION(PRODUCT_ID)
);
```

```
CREATE TABLE PRODUCT_DESCRIPTION
(
Product_ID int NOT NULL PRIMARY KEY,
Language_ID int,
Product_Name varchar(255) NOT NULL,
Product_Description varchar(255),
column1 XMLType
);
```

```
CREATE TABLE PRICE_HISTORY
(
History_ID int NOT NULL PRIMARY KEY,
ID int,
Price varchar(255),
Old_price varchar(255),
Misc varchar(255),
Date_Time date,
Username varchar(255),
FOREIGN KEY (ID)
REFERENCES PRODUCT_INFORMATION(Item_ID)
);
```

3.2 Sequence Design

These two sequences are necessary, as the two fields are primary keys. Therefore each new record has to be given a unique number which is what these sequences are achieving.

```
CREATE SEQUENCE seq_productid
MINVALUE 1
START WITH 1
INCREMENT BY 1
CACHE 10;
```

```
CREATE SEQUENCE seq_historyid
MINVALUE 1
START WITH 1
INCREMENT BY 1
CACHE 10
```

3.3 Database Trigger

//This Trigger Will fire every time a new row is inserted in to the Product Information Table.

```
create TRIGGER INSERTTRIGGER
AFTER INSERT ON PRODUCT_INFORMATION
FOR EACH ROW
BEGIN
INSERT INTO PRICE_HISTORY(History_id, ID, Old_price, Price, Misc, Date_Time, Username)
VALUES(seq_historyid.nextval, :new.Item_ID, ' ', :new.List_price, 'NEW PRODUCT', SYSDATE, USER);
END;
```

//This Trigger will fire every time an Update occurs on the Product Information Table.

```
CREATE TRIGGER UPDATETRIGGER
AFTER UPDATE OF List_Price ON PRODUCT_INFORMATION
FOR EACH ROW
BEGIN
IF :new.List_Price < :old.List_Price THEN
INSERT INTO PRICE_HISTORY(History_id, ID, Old_price, Price, Misc, Date_Time, Username)
VALUES(seq_historyid.nextval, :new.Item_ID, :old.List_Price, :new.List_price, 'Price Decreased', SYSDATE, USER);
END IF;
IF :new.List_Price > :old.List_Price THEN
INSERT INTO PRICE_HISTORY(History_id, ID, old_price, Price, Misc, Date_Time, Username)
VALUES(seq_historyid.nextval, :new.Item_ID, :old.List_Price, :new.List_Price, 'Price Increased', SYSDATE, USER);
END IF;
END;
```

//This Trigger will fire everytime a row is deleted from the Product Information table.

```
CREATE TRIGGER DELETETRIGGER
AFTER DELETE ON PRODUCT_INFORMATION
FOR EACH ROW
BEGIN
delete from price_history where ID = :old.Item_ID;
delete from product_description where PRODUCT_ID = :old.Item_ID;
END;
```


4. Implementation

Now the tables have been designed it is important to insert some information into the database to check the trigger is working correctly. The code printed below is the SQL used to insert, update and delete entries in the database tables.

As the trigger has been designed and implemented the trigger should now run once data is inserted, updated or deleted.

Screen prints of the testing process can be found in the next chapter of this report.

4.1 Insert record

//Insert a Row in to the Product description Table including the relevant XML in to the XML Type Field

```
DECLARE
XMLText1 CLOB:=
'<PRODUCT>
<PRODUCTID>1</PRODUCTID>
<LANGUAGEID>2</LANGUAGEID>
<PRODUCT_NAME>Hard Disk</PRODUCT_NAME>
<PRODUCT_DESCRIPTION>160GB Hard Disk</PRODUCT_DESCRIPTION>
</PRODUCT>';

BEGIN
INSERT INTO PRODUCT_DESCRIPTION(Product_ID, Language_ID, Product_name,Product_Description, column1)
VALUES (seq_productid.nextval, 002, 'Hard Disk','160GB Hard Disk', XMLType(XMLText1));
END;
```

//Insert Data in to the Product Information Table.

```
INSERT INTO product_information (Item_ID, Category,
Weight_Class,warranty_class,Warranty,Supplier_ID,Product_Status,List_Price,Min_Price,Catalog_URL)
VALUES (1, '1a', 'class A','W Class A','2 Years',1010,'In Stock','£125','£100','01010190');
```

4.2 Delete record

//Delete a Row from the Product Information Table.
//When this is run, the delete trigger will run and will delete the entry in the Description table and the price history table.

```
DELETE FROM product_information
WHERE Item_ID=4;
```

//Update a Field in the Product Information Table.
//The Update Trigger will run and will also update the audit table.

4.3 Update record

```
UPDATE PRODUCT_INFORMATION
SET LIST_PRICE='£1000'
WHERE ITEM_ID=8;
```

4.4 Selecting the XML

//This generates the XML requested from the tables and fields named.

```
SELECT
XMLELEMENT("ItemID", Item_ID),
XMLELEMENT("List Price", List_Price),
XMLELEMENT("Product Name", product_name),
XMLELEMENT("Product Description", Product_Description)
```

FROM product_information, product_description

5. Testing

Here are some screen prints to show the triggers working correctly when data is inserted, updated or deleted.

Workspace

```
Enter SQL, PL/SQL and SQL*Plus statements.
create TRIGGER INSERTTRIGGER
AFTER INSERT ON PRODUCT_INFORMATION
FOR EACH ROW
BEGIN
INSERT INTO PRICE_HISTORY(History_id, ID, Old_price, Price, Misc,
Date_Time, Username)
VALUES(seq_historyid.nextval, :new.Item_ID, '', :new.List_price, 'NEW
PRODUCT', SYSDATE, USER);
END;
```

Execute Load Script Save Script Cancel

Trigger created.

Workspace

```
Enter SQL, PL/SQL and SQL*Plus statements.
CREATE TRIGGER UPDATETRIGGER
AFTER UPDATE OF List_Price ON PRODUCT_INFORMATION
FOR EACH ROW
BEGIN
IF :new.List_Price < :old.List_Price THEN
INSERT INTO PRICE_HISTORY(History_id, ID, Old_price, Price, Misc,
Date_Time, Username)
VALUES
(seq_historyid.nextval, :new.Item_ID, :old.List_Price, :new.List_price, 'Pr
ice Decreased', SYSDATE, USER);
```

Execute Load Script Save Script Cancel

Trigger created.

Workspace

```
Enter SQL, PL/SQL and SQL*Plus statements.
CREATE TRIGGER DELETETRIGGER
AFTER DELETE ON PRODUCT_INFORMATION
FOR EACH ROW
BEGIN
delete from price_history where ID = :old.Item_ID;
delete from product_description where PRODUCT_ID = :old.Item_ID;
END;
```

Execute Load Script Save Script Cancel

Trigger created.

HISTORY_ID	ID	PRICE	OLD_PRICE	MISC	DATE_TIME	USERNAME
1	1	125	-	NEW PRODUCT	06-NOV-09	GILLC47
2	2	125	-	NEW PRODUCT	06-NOV-09	GILLC47
3	3	125	-	NEW PRODUCT	06-NOV-09	GILLC47
4	4	125	-	NEW PRODUCT	06-NOV-09	GILLC47
5	5	125	-	NEW PRODUCT	06-NOV-09	GILLC47
6	6	125	-	NEW PRODUCT	06-NOV-09	GILLC47
7	6	1000	125	Price Increased	06-NOV-09	GILLC47
8	4	50	125	Price Decreased	06-NOV-09	GILLC47
9	2	50	125	Price Decreased	06-NOV-09	GILLC47

9 rows selected.

6. Conclusion

During phase one of this unit the focus has been on database functionality and tools. I have become more familiar with the different tools available and have a better understanding of Triggers and XML within databases. I have also learnt about other XML technologies such as X Path and XSLT.

6.1 Recommendation

My Recommendation is that it is best to use a database server with a lot of functionality so that the developers have the best chance to manipulate the data accordingly. Oracle is a good example as it has the capability of many database tools and techniques. The tools I have talked about in this phase are all available to be used in Oracle.

7. Advanced Topic

This Advanced topic has been covered separately as a topic in its own right at the back of this report. It can be found on page 40.

8. References

W3 Schools 2009 XML Tutorial [Online]. Available:
<http://www.w3schools.com/xml/default.asp>: [accessed 9th November 2009]

W3 Schools 2009 Xquery Tutorial [Online]. Available:
<http://www.w3schools.com/xquery/default.asp>: [accessed 9th November 2009]

Database Models [Online]. Available:
<http://en.kioskea.net/contents/bdd/bddtypes.php3>: [accessed 9th November 2009]

X Path [Online]. Available: http://www.dnjonline.com/article.aspx?ID=iss25_xpath: [accessed 9th November 2009]

XML – Extensible Mark-up Language [Online]. Available: <http://www.w3.org/XML/> [accessed 9th November 2009]

Advanced and Distributed Databases

Accessing and Manipulating Data Programmatically

1. Introduction

This phase of this report is going to be briefly discussing and explaining two different types of manipulating data techniques in databases. The two main ways of manipulating data in databases is firstly to use Java to design and create a block of code to run a small program to perform actions on the database. The second way of manipulating data is to use what is known as PL/SQL to write a small program to also perform actions on the database.

The scenario for this phase is a bank account. Some privileged customers have accounts. The account table is modified according to instructions in the actions table.

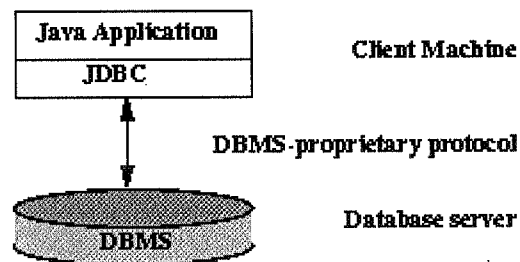
2. Background

2.1 Java JDBC/JDO

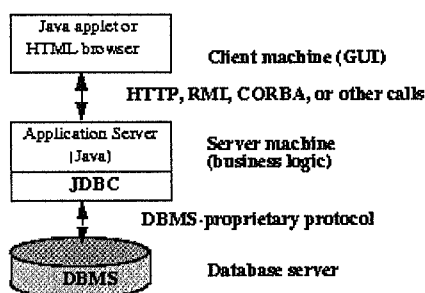
The Java JDBC is a recognised standard in the computing sector which provides connectivity between the Java programming Language and a wide range of enterprise databases. JDBC technology allows database administrators to write SQL programs for databases once and then to be run at any time on the database. This technology makes it possible to establish a connection, then to send SQL statements in the form of a program and then to process the results for the administrator.

The Java JDBC API supports both two tier and 3 tier process models for access to the database.

In the two-tier database model a database application will talk directly to the data source. This will require a JDBC driver that can communicate with the data source being accessed. A user's commands are delivered to the database, and the results of



those statements are sent back to the user. The data source may be located on another server to which the user is connected to via the network. This is often known as the client/server configuration, with the user's machine as the client, and the machine with the data source as the server.



In the three-tier database model, there is a middle tier to the model which commands are sent too from the database source. The data source will process the commands and send the results back to the middle tier, which will then send them to the user. Database administrators find the three-tier model very

attractive because the middle tier makes it possible to maintain the control needed when dealing with private company data. Another advantage is that it will simplify the deployment of applications. Finally, the three-tier architecture can provide performance advantages on the database

2.2 PL/SQL

PL/SQL is the alternative to Java JDBC. It is the language used to create a small program to execute on the database. It was introduced in to version 6 of the Oracle DBMS. It extends the concept of basic SQL with the constructs of other languages such as Pascal and Ada. The language enables the database administrator to use constructs such as different types of loops, variables, conditions and exceptions to develop their program.

There are two ways to develop a PL/SQL program. Firstly the developer can use a web-based application to connect to the database to run the program being developed. Secondly the developer may choose to use Oracle's new SQL Developer to design and execute the program. SQL developer is a graphical Interface to manage the database through tables, triggers and PL/SQL and many other database objects.

2.3 APEX

This topic is a new application released by Oracle to ease the development of front-end applications for Oracle databases. It is a web-based application for database developers to develop front-end pages for their database. All the developer will need is a web browser and limited programming experience to develop their own applications. It is possible to develop both professional and fast applications, which are secure as well. The system combines personal database productivity, ease of use and the flexibility with the qualities of an enterprise database such as security, Integrity and scalability.

3. Design & Implementation

Here is PL/SQL code, which has been designed, and developed to fulfil the requirements given in the scenero.

```

DECLARE
  CURSOR myupdate IS SELECT account_id, trans_type, new_value FROM action FOR UPDATE OF status;
BEGIN
  FOR eachrow IN myupdate
  LOOP
    eachrow.trans_type := upper(eachrow.trans_type);

    IF eachrow.trans_type = 'U' THEN
      UPDATE accounts SET bal = eachrow.new_value
      WHERE account_id = eachrow.account_id;
    END IF;

    IF SQL%NOTFOUND THEN
      INSERT INTO accounts VALUES (eachrow.account_id, eachrow.new_value);
      UPDATE action SET status = 'Account Not Found. Inserted instead.' WHERE CURRENT OF
myupdate;
    ELSE
      UPDATE action SET status = 'Account Updated.' WHERE CURRENT OF myupdate;
    END IF;

  END LOOP;

END;

DECLARE
  CURSOR myinsert IS SELECT account_id, trans_type, new_value FROM action FOR UPDATE OF status;
BEGIN
  FOR eachrow IN myinsert
  LOOP
    eachrow.trans_type := upper(eachrow.trans_type);

    IF eachrow.trans_type = 'I' THEN
      BEGIN
        INSERT INTO accounts VALUES (eachrow.account_id, eachrow.new_value);
        UPDATE action set status = 'Account Inserted' WHERE CURRENT OF myinsert;
      EXCEPTION
        WHEN DUP_VAL_ON_INDEX THEN
          UPDATE accounts SET bal = eachrow.new_value WHERE account_id = eachrow.account_id;
          UPDATE action SET status = 'Account already exists. Updated Instead' WHERE CURRENT OF myinsert;
      END;
    END IF;
  END LOOP;

END;

DECLARE
  CURSOR mydelete IS SELECT account_id, trans_type, new_value FROM action FOR UPDATE OF status;
BEGIN
  FOR eachrow IN mydelete
  LOOP
    eachrow.trans_type := upper(eachrow.trans_type);

    IF eachrow.trans_type = 'D' THEN
      DELETE FROM accounts
      WHERE account_id = eachrow.account_id;
      IF SQL%NOTFOUND THEN
        UPDATE action SET status = 'Account Number Not Found' WHERE CURRENT OF mydelete;
      ELSE
        UPDATE action SET status = 'Account Deleted' WHERE CURRENT OF mydelete;
      END IF;
    END IF;
  END LOOP;

END;

```

4. Testing

Here are some screen prints to show the PL/SQL working correctly satisfy the requirements given in the scenario for this phase.

Workspace

Enter SQL, PL/SQL and SQL*Plus statements.

```

DECLARE
  CURSOR myinsert IS SELECT account_id, trans_type,
new_value FROM action FOR UPDATE OF status;
BEGIN
  FOR eachrow IN myinsert
  LOOP
    eachrow.trans_type := upper(eachrow.trans_type);

  IF eachrow.trans_type = 'T' THEN
  BEGIN
        
```

Execute Load Script Save Script Cancel

PL/SQL procedure successfully completed.

Workspace

Enter SQL, PL/SQL and SQL*Plus statements.

```

DECLARE
  CURSOR mydelete IS SELECT account_id, trans_type,
new_value FROM action FOR UPDATE OF status;
BEGIN
  FOR eachrow IN mydelete
  LOOP
    eachrow.trans_type := upper(eachrow.trans_type);

  IF eachrow.trans_type = 'D' THEN
        
```

Execute Load Script Save Script Cancel

PL/SQL procedure successfully completed.

Workspace

Enter SQL, PL/SQL and SQL*Plus statements.

```

DECLARE
  CURSOR myupdate IS SELECT account_id, trans_type,
new_value FROM action FOR UPDATE OF status;
BEGIN
  FOR eachrow IN myupdate
  LOOP
    eachrow.trans_type := upper(eachrow.trans_type);

  IF eachrow.trans_type = 'U' THEN
    UPDATE accounts SET bal = eachrow.new_value
        
```

Execute Load Script Save Script Cancel

PL/SQL procedure successfully completed.

ACCOUNT_ID	TRA	NEW VALUE	STATUS	TIME_TAG
3	i	700	Account Updated.	26-NOV-09
6	i	20099	Account Updated.	26-NOV-09
5	d		Account Updated.	26-NOV-09
7	u	1599	Account Updated.	26-NOV-09
1	u	399	Account Updated.	26-NOV-09
9	d		Account Updated.	26-NOV-09
10	x		Account Updated.	26-NOV-09

7 rows selected.

ACCOUNT_ID	BAL
1	399
2	2000
3	700
4	6500
7	1599
6	20099

6 rows selected.

5. Conclusion

The code for the application, which has been developed, can be found on the next page. This conclusion will briefly conclude Java JDBC and PL/SQL AND relate them to real Case Studies from companies.

5.1 Java JDBC

Using Java JDBC inside an editor such as J Creator can be a very good way of developing simple program's and application's for database's. It can really help the developer to design an application, which can meet the business user requirements. The interface is usable and is fairly easy for the user to learn their way around to learn the technology. Application's built in Java for database's have proven to be reliable and robust. Java is leading the way in database application development with many developers choosing this technology. Many companies have front-end applications written in Java.

However a downside to this is that the developer has to know Java and understand Java. If a developer has not used Java much and has a limited knowledge of Java then it is best to use PL SQL to develop applications with.

An example of Java being used to develop an application can be found in the reference section of this evaluation. It is about a company who developed an application to record weather Information for an air traffic control system. They used Java to develop a graphical front end to their database to make it easier for staff to record Information.

5.2 PL/SQL

Using PL/SQL can be a really good way of developing applications for databases and is an alternative to Java JDBC. A developer who has previous experience with SQL can quite easily develop programs in PL/SQL. Developer's can write program's which can meet business user requirements and the programs, which they design and execute, can be useable to a degree as well.

Overall this evaluation concludes that Java JDBC applications are probably more useable than PL SQL applications because Java editors such as J-creator will be able to provide a better design environment for the developer to create a more usable application than PL SQL.

A typical business example where PL/SQL has been used can be found in the reference's section to this evaluation. It is a case study of a company who developed an application to check for suspicious customer activities online with a bank's website to be detected. The full case study is available in the references.

5.3 Recommendation

My general recommendation here is that if the developers have experience in designing and creating JAVA applications then I recommend using Java JDBC to create database driven Applications. On the other hand if the developers have a limited knowledge of JAVA and are unsure then I recommend using PL/SQL as this is a programming language which is used to write simple database applications in the form of SQL.

6. References

Oracle Online [Online]. Available: <http://java.sun.com/docs/books/tutorial/jdbc/overview/index.html>: [accessed 30th November 2009]

What is APEX [Online]. Available:
http://www.oracle.com/technology/products/database/application_express/html/what_is_apex.html: [accessed 30th November 2009]

PL/SQL Technology Centre [Online]. Available:
http://www.oracle.com/technology/tech/pl_sql/index.html: [accessed 30th November 2009]

PL SQL Case Study [Online]. Available:
<http://www.theoptimum.net/new/datawh-bi-case-studies.asp>: [Accessed 30th November 2009]

Java Case Study [Online]. Available:
http://www.oracle.com/technology/tech/java/java_db/pdf/javadb_use_case_edited_2.pdf: [Accessed 30th November 2009]

Advanced and Distributed Databases

Improving Data Access by Data Distribution and Replication

1. Introduction

This subject area within databases is all about accessing data within different databases, which may be on different networks in different countries. SQL allows database developers to distribute data to different servers throughout a business.

The need for this is very important, as businesses may be international based with sites in different countries. Therefore their staff will need to be able to access company data stored in other database systems in other countries.

The scenario for this phase is a company rapidly expanding its international business. There are now many locations in four main regions of the world. However its HR and order entry systems are still centralised at one site.

In order to improve this setup the company has decided to distribute and or replicate where appropriate across its international sites.

2. Background

2.1 Data Replication

This is the process of sharing information to ensure consistency between separate database systems. There are two types of data replication. The first type is called active replication. This is performed when processing the same request over and over again. In passive replication each single request is processed one at a time and then its state is transferred to the database system.

2.2 Key Advantages of Replication

- a. Availability - Replication improves the availability of applications because it provides them with alternative data access options. If one database becomes unavailable then the users can continue to query or even update the remaining databases.
- b. Performance – Replication provides fast, local access to shared data because it balances activity over multiple sites. Some users will be able to access one database and others will access other databases. Therefore the load on one database is spread across a number of databases on the network.
- c. Disconnected Computing – A materialised view is a complete or partial view of a target table from a single point in time. Materialised views enable users on a subset of a database while disconnected from the central database.
- d. Network load Reduction – It can also be used to distribute data over

multiple regional locations. The applications can access various servers instead of accessing one central server. This configuration can reduce network load in a big way.

2.3 Data Distribution

This is the process that manages a single copy of all the data and all the database objects. Data distribution in the context of databases is the term given to a database, which is spread over a network. This data can be stored in different databases. Each database in a distributed setup has connections to other databases so that other databases can be queried and data is replicated on other databases.

2.4 Database Links

A database link is a pointer to another database, which can be on the same network or in a different country. The following is an example of SQL to create a database link in to a database system.

```
CREATE PUBLIC DATABASE LINK student3 CONNECT TO blogs IDENTIFIED BY USING  
'student3.solent.ac.uk'
```

This would mean that anybody who is connected to this database would be able to use this link to query the database called student3.

Aso general

2.5 Distributed Queries

The database where the query is sent from would break down the query in to a number of remote queries, which then sends the remote nodes ready to be executed. The remote node then executes the query and sends the results back to the local node. The local nodes then perform any necessary processing and return the results to the local database system.

3. Design & Implementation

3.1 Task Two Code

```
CREATE VIEW HR_employees as
SELECT employees.employee_id, first_name, last_name, department_name, state_province, country_name, job_title,
max_salary
FROM hrd.employees@student3, hrd.departments@student3, hrd.locations@student4, hrd.countries@student4,
hrd.jobs@student3
WHERE employees.department_id = departments.department_id
AND departments.location_id = locations.location_id
AND locations.country_id = countries.country_id
AND jobs.job_id = employees.job_id
UNION
```

```
SELECT employees.employee_id, first_name, last_name, department_name, state_province, country_name, job_title,
max_salary
FROM hrd.employees@student4, hrd.departments@student4, hrd.locations@student4, hrd.countries@student4,
hrd.jobs@student4
WHERE employees.department_id = departments.department_id
AND departments.location_id = locations.location_id
AND locations.country_id = countries.country_id
AND jobs.job_id = employees.job_id
UNION
```

```
SELECT employees.employee_id, first_name, last_name, department_name, state_province, country_name, job_title,
max_salary
FROM hrd.employees@student5, hrd.departments@student5, hrd.locations@student4, hrd.countries@student4,
hrd.jobs@student5
WHERE employees.department_id = departments.department_id
AND departments.location_id = locations.location_id
AND locations.country_id = countries.country_id
AND jobs.job_id = employees.job_id
ORDER BY max_salary DESC;
```

3.2 Snapshot Log

This is the script which generates the Trigger to take a snapshot when the departments table is updated and inserts the new row in to the snapshot table.

```
create or replace trigger departments_update
before update of department_name on departments
for each row
begin
insert into
DEPARTMENTS_SNAPSHOTLOG(log_id,updated_master_primary_key,updated_master_timestamp,refresh_snapshot_timest
amp)
values(log_id_seq.nextval, :old.department_id, SYSDATE, NULL);
end;
```

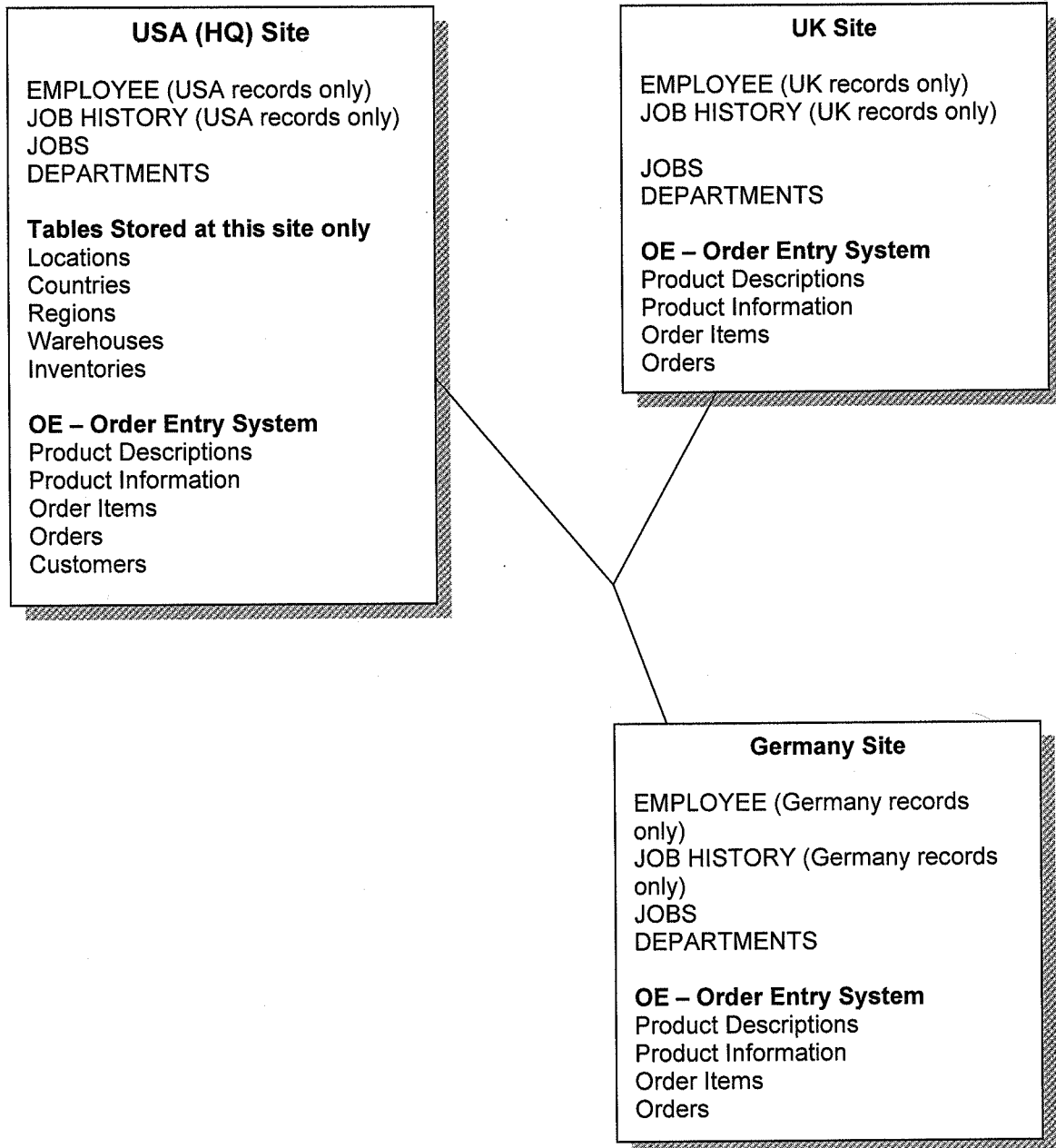
Department Snap Shot Table

```
CREATE TABLE DEPARTMENTS_SNAPSHOTLOG
(
LOG_ID NUMBER(5,0),
UPDATED_MASTER_PRIMARY_KEY NUMBER(6,0),
UPDATED_MASTER_TIMESTAMP DATE,
REFRESH_SNAPSHOT_TIMESTAMP DATE,
PRIMARY KEY ("LOG_ID")
)
```

Department Database Table

```
CREATE TABLE DEPARTMENTS
(
DEPARTMENT_ID NUMBER(4,0),
DEPARTMENT_NAME VARCHAR2,
MANAGER_ID NUMBER(6,0),
LOCATION_ID NUMBER(4,0)
)
```

4. Distributed Database Design Model



4.1 Explanation for Proposed Model

The proposed model has been designed with a few requirements outlined by the company in the scenario.

These requirements are:

- a. An order transaction is completed regardless of whether the site is connected to databases at other sites.
- b. Update of orders to other sites does not need to be instantaneous, although consistency of product and price information is a high priority.
- c. The System must be designed so that distributed queries at the HQ can be run on the database. This will enable managers to easily retrieve information (such as a list of employees) for specific sites or for the entire company.

There are specific tables that the order entry system must distribute across all three sites. This is to ensure that if the connections between the three sites go down, the company across all three sites can still place orders and operate efficiently. These tables are the product Information, Product descriptions, orders Items and orders.

The customer's table is going to be fully replicated across all three sites. This is because each site needs access to the customer's table in order to place orders with the correct customer.

This leaves the last two tables, which is the inventories and warehouses table. This is going to be portioned at the USA (HQ) site only. This is because they are not necessary to replicate across to all three sites. If the database links go down the separate site databases can hold any updates until the full linked system is resumed again. Then it can update these two tables with the new information.

The other side of the system is the HR Database. This has already been implemented across the company.

5. Testing

Here are some screen prints to show the select statements across the three different sites and the snapshot trigger and table as well to ensure that they are correctly working correctly to satisfy the requirements given in the scenario for this phase.

Workspace

Enter SQL, PL/SQL and SQL*Plus statements.

```
CREATE VIEW HR_employees as
SELECT employees.employee_id, first_name, last_name,
department_name, state_province, country_name, job_title, max_salary
FROM hrd.employees@student3, hrd.departments@student3,
hrd.locations@student4, hrd.countries@student4, hrd.jobs@student3
WHERE employees.department_id = departments.department_id
AND departments.location_id = locations.location_id
AND locations.country_id = countries.country_id
AND jobs.job_id = employees.job_id
UNION
```

Execute

Load Script

Save Script

Cancel

View created.

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	DEPARTMENT_NAME	STATE_PROVINCE	COUNTRY_NAME	JOB_TITLE	MAX_SALARY
100	Steven	King	Executive	Washington	United States of America	President	4000
101	Neena	Kochhar	Executive	Washington	United States of America	Administration Vice President	3000
102	Lex	De Haan	Executive	Washington	United States of America	Administration Vice President	3000
145	John	Russell	Sales	Oxford	United Kingdom	Sales Manager	2000
146	Karen	Partners	Sales	Oxford	United Kingdom	Sales Manager	2000
147	Alberto	Errazuriz	Sales	Oxford	United Kingdom	Sales Manager	2000
148	Gerald	Cambrault	Sales	Oxford	United Kingdom	Sales Manager	2000
149	Eleni	Zlotkey	Sales	Oxford	United Kingdom	Sales Manager	2000
108	Nancy	Greenberg	Finance	Washington	United States of America	Finance Manager	1600
205	Shelley	Higgins	Accounting	Washington	United States of America	Accounting Manager	1600
114	Den	Raphaely	Purchasing	Washington	United States of America	Purchasing Manager	1500
150	Peter	Tucker	Sales	Oxford	United Kingdom	Sales Representative	1200
151	David	Bernstein	Sales	Oxford	United Kingdom	Sales Representative	1200
152	Peter	Hall	Sales	Oxford	United Kingdom	Sales Representative	1200
153	Christopher	Olsen	Sales	Oxford	United Kingdom	Sales Representative	1200
154	Nanette	Cambrault	Sales	Oxford	United Kingdom	Sales Representative	1200
155	Oliver	Tuvault	Sales	Oxford	United Kingdom	Sales Representative	1200
156	Janette	King	Sales	Oxford	United Kingdom	Sales Representative	1200
157	Patrick	Sully	Sales	Oxford	United Kingdom	Sales Representative	1200
158	Allen	Martin	Sales	Oxford	United Kingdom	Sales Representative	1200



6. Conclusion

This chapter has focussed on the importance of data distribution and data replication. To begin with it was difficult to understand what tables to use from different sites so it took me a while to fully understand chapter three correctly. However now I have a good understanding of connecting databases together to use data.

7. References

About.com - Databases, 2010. *Data Replication* [online]. Available: <http://databases.about.com/cs/sqlserver/a/aa041303a.htm> [accessed 16th March 2010]

IBM, 2010. Advantages of *Data Replication* [online]. Available: <http://publib.boulder.ibm.com/infocenter/idshelp/v10/index.jsp?topic=/com.ibm.admin.doc/admin622.htm> [accessed 16th March 2010]

Data Replication, 2010. *Data Replication* [online]. Available: http://www.blueclaw-db.com/broad_interest/database-replication.htm [accessed 16th March 2010]

Data Replication, 2010. *Data Replication* [online]. Available: http://www.blueclaw-db.com/broad_interest/database-replication.htm [accessed 16th March 2010]

Database distribution, 2010. *Data Replication* [online]. Available: <http://msdn.microsoft.com/en-us/library/ms183524.aspx> [accessed 18th March 2010]

Data Replication, 2010. *Data Replication* [online]. Available: <http://www.topbits.com/database-replication.html> [accessed 25th March 2010]

Advanced and Distributed Databases

Multi – Dimensional Modelling and Analysis for Decision Support

1. Introduction

This chapter will discuss the tools and methods associated with this theme within databases. This theme was largely about using databases to make decisions with businesses and organisations.

Within a large organisation there is often two large databases. The first database is the operational database with all the day to day important Information. This is how the business operates. However often there is a second database, which is used to store history of the company. This type of database is called an OLAP database. This can often store anything about the business, transactions, customers, orders, sales and many, many more. This type of database just keeps getting larger with more and more Information inserted in to it. The bigger the database gets, then the more useful it is to provide business answers. As a result over time the need for more storage is needed and so the hardware is constantly being upgraded.

The scenario for this phase was based on a company who wishes to use their company data, which is being collected to provide useful results and information to aid decision making in the future for the company.

2. Background

2.1 Different OLAP Tools

Generated by Excel from Oracle data source

Sum of AMOUNT	CUSTID	101	102	103	104	105	106	107	108
100860	100	350	35	?	0	3000	1750	440	0
100861	100	485	0	2300.5	450	610	4500	4584	180
100870	100	292.5	8.4	3306	140	0	1400	1400	280
100871	100	0	0	5600	0	846.8	?	0	250
100890	100	50	58	0	16	0	29000	464	0
101860	100	0	0	0	0	24	2400	1200	0
101863	100	0	0	0	0	1500	2500	900	0
102130	100	1703.4	0	0	34	340	340	34	0
200376	100	2400	0	0	24	240	480	2.4	0
200380	100	0	0	0	0	400	1200	0	600

Generated from ISQLPlus

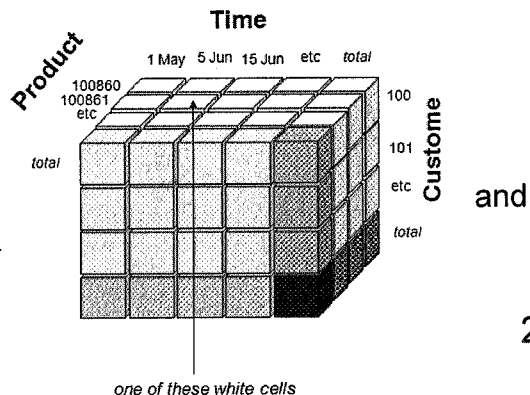
CUSTID	PRODID	SUM(AMOUNT)
100	100860	350
100	100861	485
100	100870	292.5

First three rows from SELECT custid, prodid, SUM(amount) FROM saleswh.sales GROUP BY custid, prodid;

There are many different types of tools developers can use to get useful results out of the OLAP database. One of these tools can be pivot dimensions. This is a function within a spreadsheet such as MS Excel which allows a set of results from the database to be Inserted and then the spreadsheet takes these results and turns it in to a table so the Information can be read

easily. This type of method is called cross-tabular analysis and is a very common tool with decision support. It allows the developers to see the overall result of the SQL record set very easily.

Another type of tool, which can be used for decision support, is the drill down approach. This can be very useful if the developer wants to 'drill down' to a greater level of each individual customer. The best way to understand this is to look at it like a cube. Where product is the width, time is the length the number of customer's is the height. An example of the SQL which could be



Chris Gillman
Advanced and Distributed Database's

generated could be the following:

```
Select cust_id, time_id, SUM (no_sold)
FROM sales
GROUP BY rollup (prod_id)
```

Another type of tool which can be very useful to aid decision support is the rank measures tool. This tool is used when an SQL script is developed and used. The SQL consists of the 'rank' keyword, which ranks the SQL in an order set by the developer. An example of the SQL code which can be used is the following:

```
SELECT prod_id, description, sum(amount), RANK() OVER (ORDER BY sum(amount)
DESC) AS rank
FROM saleswh.product p, saleswh.sales s
WHERE p.prod_id = s.prod_id
GROUP BY p.prod_id, p.description;
```



3. Design & Implementation

SQL Statement

```
select cust_first_name, cust_last_name, quantity_sold, sum(amount_sold) as 'Total Amount',
countries.country_id, countries.country_name
FROM customers, sales, countries
WHERE customers.country_id = countries.country_id
AND customers.cust_id = sales.customer_id
AND ROWNUM <=50
GROUP BY customers.cust_id
ORDER BY amount_sold DESC
```

Showing the top 50 customers bringing in the most revenue which also identifies which country they are from.

4. Dimension Modelling

This type of modelling is a concept used by many data warehouse designers to build their data warehouse. There are two types of tables in a dimension model. First of all there are Fact tables. These contain the facts and measurements of the business. Then the second type of table is the dimension table. These contain the context of measurements, for example the context of dimensions of which the facts are calculated.

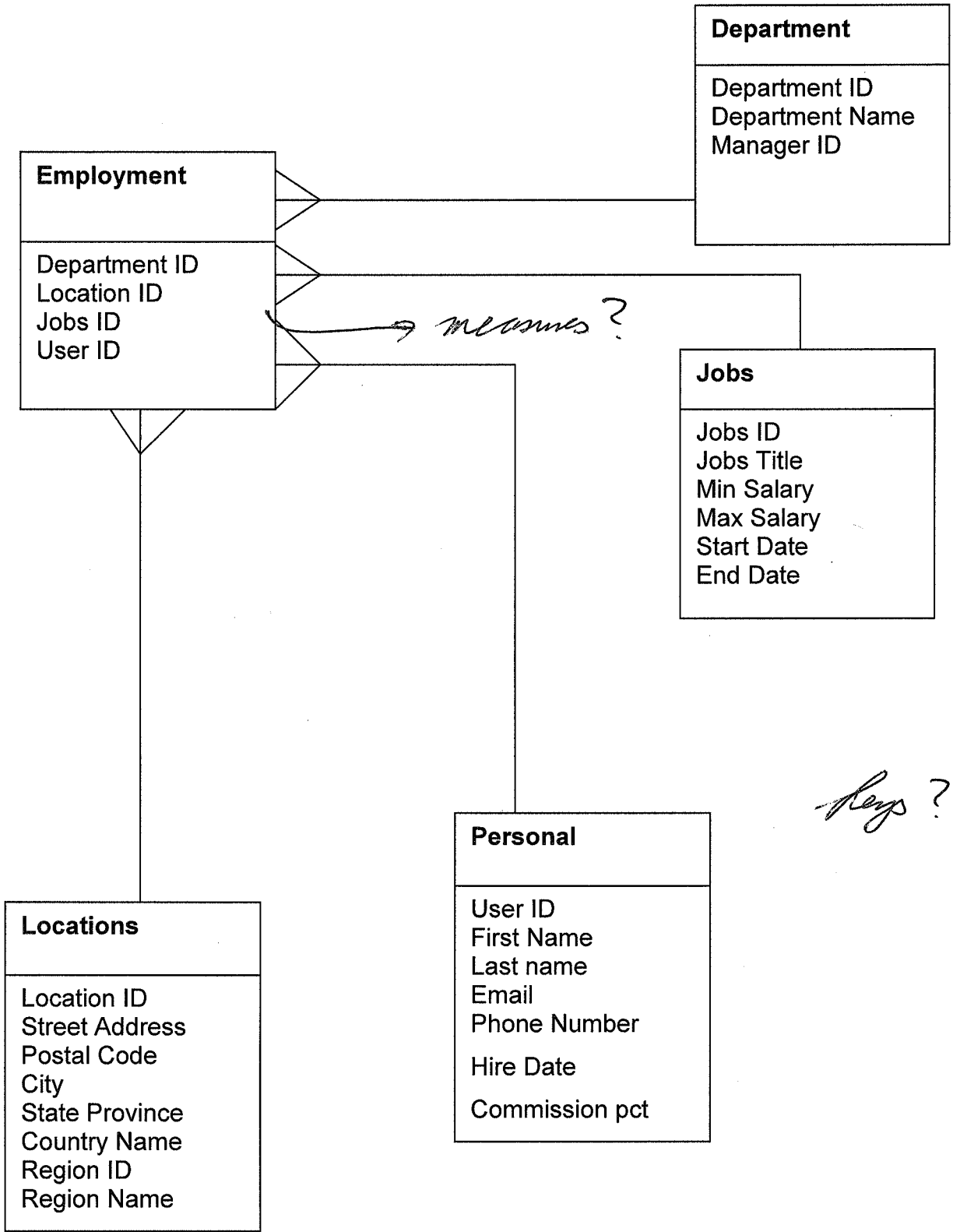
Another name for the Dimension Model is the star-join schema. Database designers use this name because the diagram looks like a star as all the dimensions are around the edge with the fact table in the middle.

The Dimension tables only have a single join that attaches them to the fact table. The fact table has multiple joins linking to every Dimension table.

Over on the next page there is a Dimension Model for the HR database.



4.1 Dimension Model



5. Testing

Here are some screen prints to show the development of my useful result set.

3	Sum of Number Sold	Customer ID								
4	Product ID	100	101	102	103	104	105	106	107	108
5	100860	10	1	471	0	100	50	10	0	0
6	100861	11	0	51	10	20	100	102	4	0
7	100870	105	3	1030	50	0	500	500	100	0
8	100871	0	0	1000	0	153	500	0	50	100
9	100890	1	1	0	2	0	500	8	0	0
10	101860	0	0	0	0	1	100	50	0	200
11	101863	0	0	0	0	150	200	100	0	0
12	102130	501	0	0	10	100	100	10	0	0
13	200376	1000	0	0	10	100	200	1	0	200
14	200380	0	0	0	0	100	300	0	0	150
15										

Customer ID	Product ID	Number Sold
100	100860	10
100	100861	11
100	100870	105
100	100890	1
100	102130	501
100	200376	1000
101	100860	1
101	100870	3
101	100890	1
102	100860	471
102	100861	51
102	100870	1030
102	100871	1000
103	100861	10
103	100870	50
103	100890	2
103	102130	10

Last Name	Australia	Canada	France	Germany	Japan	Singapore	Spain	The Netherlands	United Kingdom	USA	Grand Total
Abbassi							1232.99		1237.31		2470.3
Abbey										1232.16	1232.16
Barlow							2465.15				2465.15
Callihan										1232.16	1232.16
Dally										1237.31	1237.31
Drumm										1237.31	1237.31
East	1232.99										1232.99
Edmond										1232.99	1232.99
Everley								1237.31			1237.31
Gatewood	1232.16										1232.16
Gilmour										1237.31	1237.31
Hogan				1232.16							1232.16
Kayden				1232.16							1232.16
Kiker										1232.16	1232.16
Kipp										1237.31	1237.31
Lehman					1232.16						1232.16
Lickey									1232.99		1232.99
Lin										1232.16	1232.16
Lincoln										2470.3	2470.3
Linsicome										1232.16	1232.16
Lipp										2470.3	2470.3
Lowers										1237.31	1237.31
Lucas							1232.16				1232.16
Mahood				1237.31							1237.31
Majors										1237.31	1237.31
Markin				1237.31							1237.31
Nance										1232.16	1232.16

6. Conclusion

During phase four, the topic has focussed on the different tools and methods, which can be used to aid, decision support. This is such a large area of databases as there are a lot of very useful tools, which can be used to extract useful Information out of the database.

I have an understanding of these tools and have showed evidence of one of these tools in chapter three of this phase

7. References

Microsoft Library [online]. Available: <http://msdn.microsoft.com/en-us/library/ms177410.aspx> 11th February 2010

Simple Talk [online]. Available: <http://www.simple-talk.com/sql/t-sql-programming/creating-cross-tab-queries-and-pivot-tables-in-sql/> 11th February 2010

Data Warehousing Review Available: http://www.dwreview.com/OLAP/OLAP_Comparison.html 11th February 2010

Dimension Modelling [online]. Available: <http://it.toolbox.com/blogs/rossgoodman-on-bi/dimensional-modeling-facts-2288612> March 2010

Dimension Modelling [online]. Available: <http://www.oracle.com/technology/products/warehouse/pdf/Benefits%20of%20a%20multi-dimensional%20model.pdf>: March 2010.

Advanced and Distributed Databases Mining Databases for Decision Support

1. Introduction

The purpose of this phase is to produce an analysis to aid decision making for the management of a company. The type of data mining, which is going to be used, is the Market Basket Analysis Method. This will be discussed later in this report. However before the analysis can start the data has got to be prepared. So the following step by step guide is set out in sections to show how the data was prepared ready for data mining.

2. Background

As an overview this is a tool where data mining analysts use to draw comparisons and rules out of the data. It is best described as thinking about a shop whether it's a high street shop or an online shop, it makes no difference. The IT Infrastructure within a business could well be setup to record all orders/transactions in an OLAP database for further research and analysis to be done. Once they have this data they can extract the data from the database using SQL and perform various techniques on the data. One of these techniques is the Market Basket Analysis.

2.1 Market Basket Analysis

As described above this is a tool which generally asks questions such as: If the customer buys Product, then how likely is it that they will buy product B. An example to this is: If the customer buys a printer then how likely is it that they will buy paper and ink cartridges as well. With the help of a tool like Weka the rules and answers can be bought up very easily. This can be done manually as well. By extracting the data from the database and copying and pasting this in to the spreadsheet the data can be pivoted using the pivot table function and the data can be read and counted down. For example see below:

2.2 Sample Market Basket Analysis

	Prod_1	Prod_2	Prod_3	Prod_4
Cust_1	Yes	Yes	?	Yes
Cust_2	?	yes	Yes	Yes
Cust_3	yes	?	Yes	Yes

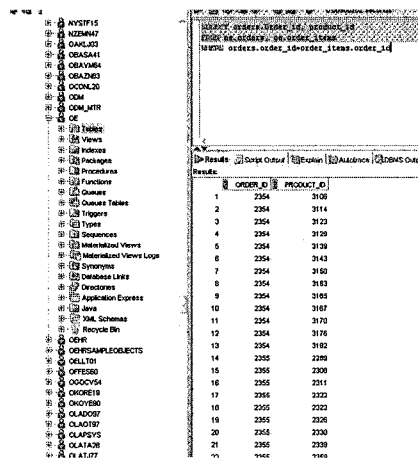
So as the table shows, customer 1 bought items 1,2,4. So If the developer is asked by Management how often is item 1 and 2 bought together then the answer is once. However if Management asked how often is item 3 and 4 bought together then the answer is 2.

Information like this passed on to Management can help with displaying the shop to aid sales and customer flow.

An actual example can be taken from the example showed earlier in this report.

3. Design

If the database is vast with thousands of customers and products then the ideal solution is to use a tool like Weka. Performing the steps outlined earlier in this report Weka will read in the CSV file. Then when the Associate tab is clicked and then the start button is also clicked Weka will match rows automatically in the data set automatically and return the results.



3.1 Stage One

This is a screen print of SQL developer. This was used to connect to the Oracle database and to retrieve the Information. SQL Statements were used to retrieve the Information, which was needed. The Script which was used was:

```
SELECT orders.Order_id, product_id
FROM oe.orders, oe.order_items
WHERE orders.order_id=order_items.order_id
```

3.2 Stage Two

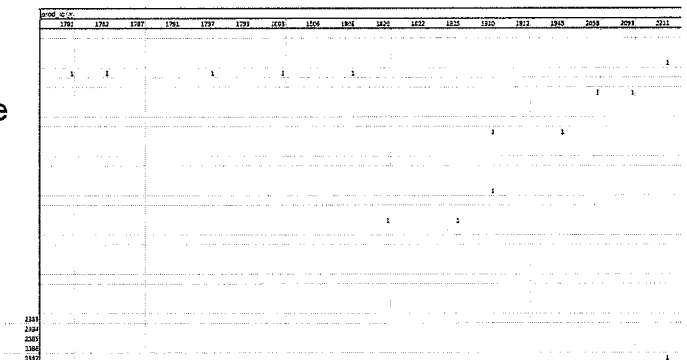
The next stage was to use this data which the SQL statement had retrieved. So the best way forward was to copy and paste the entire contents of this result set in to MS Excel.

The top row was left blank so the titles could be typed in. On the right, there is the result set in Excel. A third column was added which would end up to be the data column once it has been turned in to a Pivot table within Excel.

Order ID	Prod ID	
3	2354	3106
4	2354	3114
5	2354	3122
6	2354	3129
7	2354	3135
8	2354	3143
9	2354	3150
10	2354	3163
11	2354	3165
12	2354	3167
13	2354	3170
14	2354	3176
15	2354	3182
16	2355	2289
17	2355	2308
18	2355	2311
19	2355	2322
20	2355	2323
21	2355	2326
22	2355	2355
23	2355	2359
24	2355	2359
25	2356	2284
26	2356	2274
27	2356	2295
28	2356	2299
29	2356	2308
30	2356	2311
31	2356	2318
32	2356	2329
33	2357	2211

3.3 Stage Three

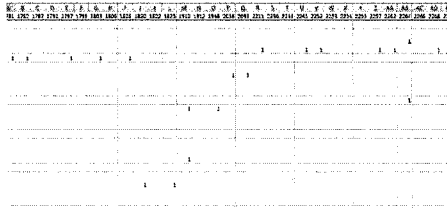
The next stage was to turn this raw data result set in to a Pivot table. To do this the raw data was highlighted and the Excel function was run. Once the wizard was finished it turn out to look like the Image on the left. However this needed to be edited down and simplified as this was going to be saved as a .CSV file. Therefore there was not allowed to be any Excel Formatting.



of the Pivot table including the Row headings (Product_ID) but not the left Chris Gillman Advanced and Distributed Database's

column (Order_No). Then the highlighted area was copied and was pasted in to a new spreadsheet. The columns were adjusted to size and the outcome was the following below.

The next stage was to replace all the blank cells with a '?' and all the '1's' with a 'yes'. Then finally to add a blank row under the product ID's
The result can be seen below.

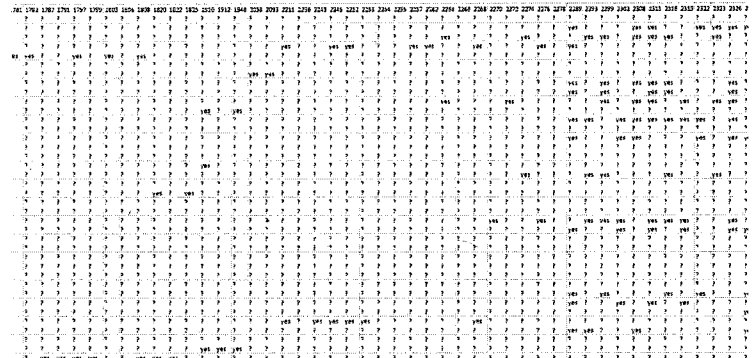


The next stage was to replace all the blank cells with a '?' and all the '1's' with a 'yes'. Then finally to add a blank row under the product ID's
The result can be seen below.

3.4 Stage Five

This is the final result set which is ready to be saved as a CSV file for the Data Mining technique.

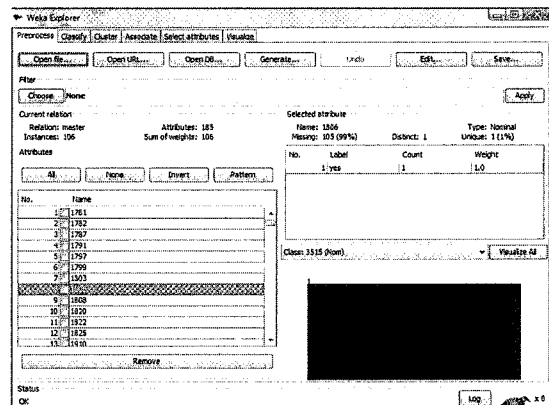
The final task, which had to be completed, was to remove any other sheets in this workbook. Therefore the raw data sheet and the Pivot table sheet were deleted leaving just this table.



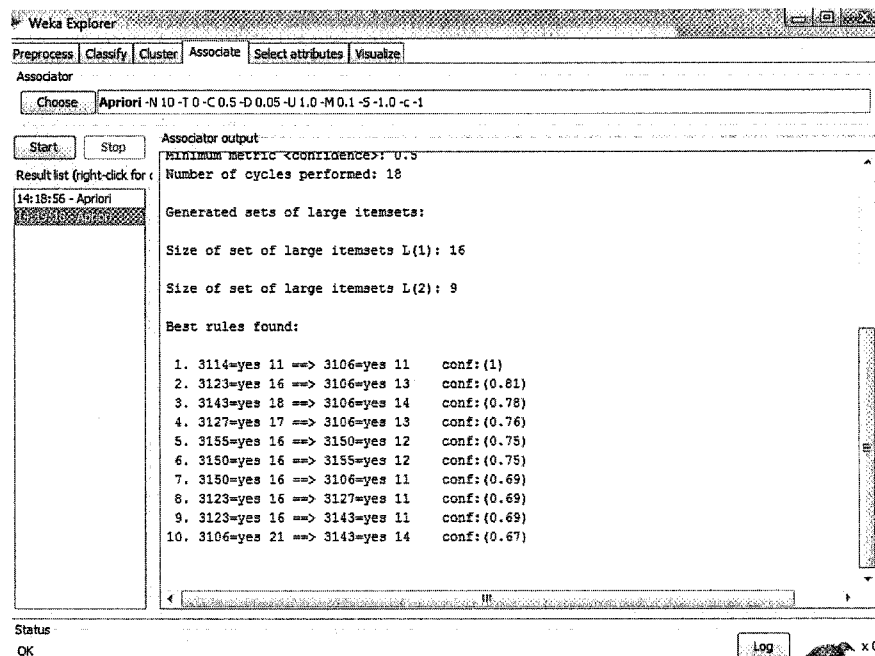
4. Implementation

The data is now ready to be used with a Data Mining tool. The tool, which was used was called 'Weka'. This is a tool, which is practically for Data Mining. The tool has been referenced in the References of this report and can be downloaded for non commercial use in a Java and non Java edition.

Once Weka has been downloaded and Installed. It was then opened and the CSV which had been created. The weka interface then looked like the following. Weka can be used for many different types of Data Mining. It is a powerful tool, which can be used to get lots of different types of analysis out of the data. In this case Weka has been used to help with the Market Basket Analysis.



5. Testing – Confidence levels in Weka



A screen print of the results screen in Weka is shown below:

6. Tools and Technologies

This section of the report will focus on evaluating the different technologies, tools and methods associated with data mining. As discussed earlier in this report data mining is incredibly useful to find rules and patterns to present to management as it will assist with the layout of the store and position of the stock. This will also benefit and improve the service towards customers as they will be able to find what they are looking for very quickly. One of the tools associated with data mining is Weka which has already been discussed and used in this report.

6.1 Association Rules

One method involved with data mining is Association rules. These are really important as they discover how products are linked together. Basically they ask the question of how likely two products or more sell together. Then a tool such as Weka can perform this task and would output rules based on percentages which would show how likely that two products would sell together.

6.2 Product Hierarchies

Another method involved with data mining is Product Hierarchies. Every Item in a high street shop or in an online shop would have a SKU, which stands for stock keeping unit. Every SKU depending on the product falls in to a hierarchical category which starts really general and level by level gets more specific to describe the item.

6.3 Confidence level

Tools like Weka calculate the confidence level. Which means how certain it is that an item will sell. For Instance tools can work out the probability of item A selling on its own. But can also work out the probability of item A and B selling together. If the probably is high for A and B selling together this would be Information for management because based on selling trends these two products go together and will probably continue to sell if placed together in the shop.

6.4 Drill-Down Analysis

Drill down analysis applies to data mining to go deeper and to extract the data required. This is in particular for large databases. The process starts by considering some simple break downs of the data such as gender, region etc. Various tools such as histograms, tables can be used for each group to extract useful data. Once this initial analysis has taken place a further drill down may be necessary for example a company may want to review the data from males from a specific county. So instead of just looking at data from both genders for all regions the company can drill down further to get more specific data from the database.

7. Conclusion

This phase has been really interesting to learn how to use a database to extract useful data from the record set. The market basket analysis is a very useful tool to use as it looks look at results of how popular certain products are when bought together.

Weka data mining tool is also a useful tool when dealing with market basket Analysis as it compares the result set and returns different confidence level matches. Weka can handle many more data mining techniques so it is very much worth using with data mining.

8. References

University of Waikato, 2008. *Weka Data Mining Tool* [online]. Available: <http://www.cs.waikato.ac.nz/ml/weka/> [accessed 11th March 2010]

Statsoft tools, 2008. *Data Mining Techniques* [online]. Available: <http://www.statsoft.co.uk/textbook/stdatmin.html> [accessed 11th March 2010]

Data Mining, 2008. An Overview of Data Mining Techniques [online]. Available: <http://www.thearling.com/text/dmtechniques/dmtechniques.htm> [accessed 11th March 2010]

Oracle Data mining, 2010. An Overview of Data Mining Techniques [online]. Available: <http://www.oracle.com/technology/products/bi/odm/index.html> [accessed 18th March 2010]

Oracle Data mining, 2010. An Overview of Data Mining Techniques [online]. Available: <http://www.oracle.com/technology/products/bi/odm/odminer.html> [accessed 23th March 2010]

Data mining, 2010. An Overview of Data Mining Techniques [online]. Available: <http://blogs.oracle.com/datamining/> [accessed 23th March 2010]

Advanced and Distributed Databases

X Query – Advanced Topic

1. Introduction

X Query is a language used to query XML data within database tables. Where as SQL is used to query data in database tables, X Query is used in the same way to query the XML found in database tables. It is used to find and extract elements from the XML data. It is designed to query XML data within a database and not just XML files. In fact any data source which appears as XML can be queried using X Query. It was built and is based on X Path expressions and is supported by all major databases. It is also a W3c recommendation so it's starting to be used more and more widely.

X Query provides a framework to efficiently and easily extract Information from relational databases, files and other data sources. X Query also allows you to construct XML data by using the result of a query.

With X Query the developer has the ability to collect Information from multiple documents in a single query.

X Query is compatible with several w3C Standards such as XML, XSLT, X Path, XML Schema. It became a W3c recommendation back in 2007.

2. Background

2.1 X Query within Oracle

X Query also allows you to construct XML data by using the result of a query. With Oracle's X Query the developer can view and query a relational database table as just another XML data source. One of X Query's most useful features is its ability to run a single query on multiple data sources. For example, you can run a query that combines an incoming purchase order in XML format, an archive of catalogue data in XML format, and an inventory system held in a relational database. All three examples can be queried using X Query.

2.2 Examples of its use

Firstly X Query can be used to extract information from a webservice. Secondly XQuery can be used to generate summary reports. Then it can be used to transform XML data to XHTML. Then finally it can also be used to search web documents for Information.

A query in XQuery is a statement, which reads a sequence of XML fragments and also returns a sequence of XML fragments. The principal forms of X Query expressions are path expressions, element expressions, list expressions and data type expressions.

3. Scenario for Advanced Topic

There is a need within the business to implement an X Query expression to generate XML from a specific table held within the OE database.

The business would like to use X Query to retrieve the contents of the product descriptions table and export this as XML so that this XML can be read by another company to display this data and style it on their own website. An example of this could be another company wishing to sell products and would ideally want to show the products on their own website. This company holding the OE database would need a web service so that other companies can connect and search the XML..

4. Design

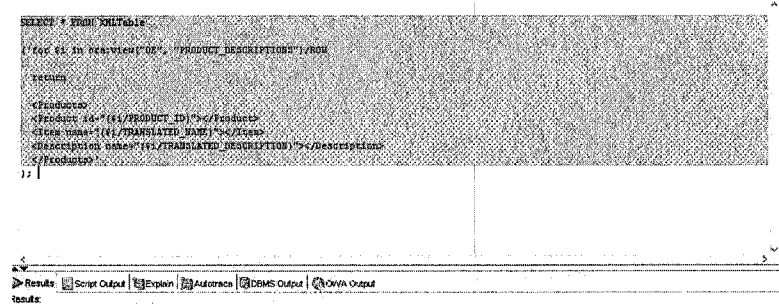
```
SELECT * FROM XMLTable
```

```
('for $i in ora:view("OE", "PRODUCT_DESCRIPTIONS")/ROW
```

```
return
```

```
<Products>
<Product id="{ $i/PRODUCT_ID }"></Product>
<Item name="{ $i/TRANSLATED_NAME }"></Item>
<Description name="{ $i/TRANSLATED_DESCRIPTION }"></Description>
</Products>'
);
```

5. Testing



Evaluates?

6. Conclusion

X Query is a very useful query engine which is built in to database systems. It has the capability to either retrieve XML from the XML type column in a database table or as seen above the ability to generate XML from a database table. I highly recommend X query as it is a really good way of generating XML so that the XML can be used as website content.

7. References

W3 Schools, 2009. *X Query* [online]. Available:

http://www.w3schools.com/xquery/xquery_summary.asp [accessed 18th March 2010]

Oracle, 2009. *X Query* and XPath [online]. Available:

http://download.oracle.com/docs/cd/E13159_01/osb/docs10gr3/consolehelp/proxyeditors.html [accessed 18th March 2010]

WSc, 2010. *X Query* [online]. Available: <http://www.w3.org/TR/xquery/> [accessed 18th March 2010]