# 7CI019 Medcruise Case Study

Student number: 1007022

Module code/title: 7CI019 Database Technologies

Submission Date: 14/04/2015

Table of Contents

[7CI019 Medcruise Case Study 1](#_Toc417579936)

[Portfolio Part One 2](#_Toc417579937)

[Q1 - Design 2](#_Toc417579938)

[Assumptions 3](#_Toc417579939)

[Set of Tables 3](#_Toc417579940)

[Q2 – Populate your database 5](#_Toc417579941)

[Q3 – SQL Queries 6](#_Toc417579942)

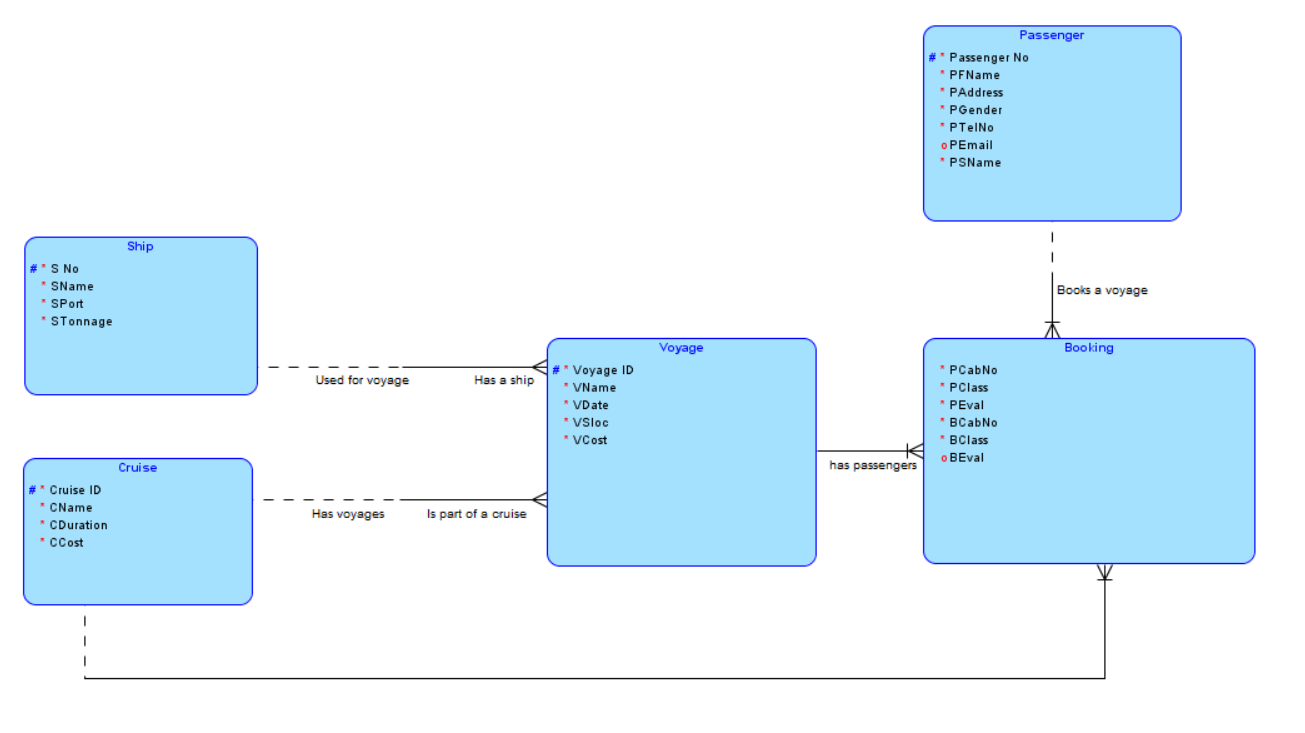
[Q4 – View 11](#_Toc417579943)

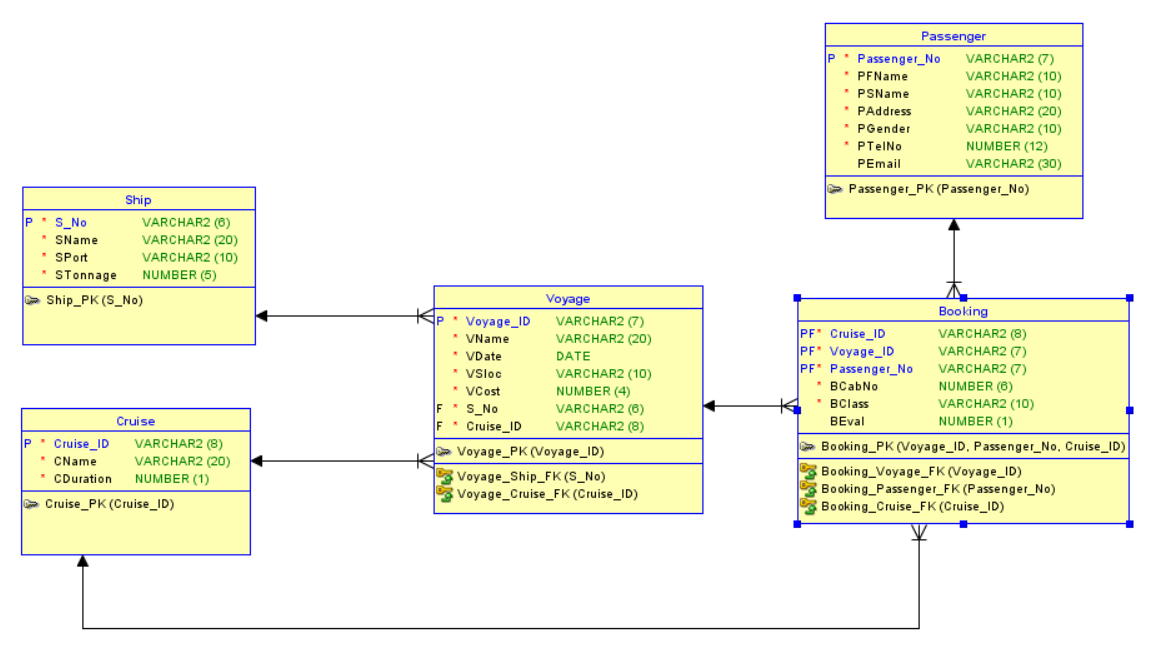
[Q5 – Triggers 14](#_Toc417579944)

[Q6 – Report 16](#_Toc417579945)

# Portfolio Part One

## **Q1 - Design**





### Assumptions

## Set of Tables

-- Generated by Oracle SQL Developer Data Modeler 4.0.3.853

-- at: 2015-04-22 20:31:31 BST

-- site: Oracle Database 12c

-- type: Oracle Database 12c

CREATE TABLE Booking

(

Cruise\_ID VARCHAR2 (8) NOT NULL ,

Voyage\_ID VARCHAR2 (7) NOT NULL ,

Passenger\_No VARCHAR2 (7) NOT NULL ,

BCabNo NUMBER (6) NOT NULL ,

BClass VARCHAR2 (10) NOT NULL ,

BEval NUMBER (1)

) ;

ALTER TABLE Booking ADD CHECK ( BClass IN ('First', 'Second', 'Third')) ;

ALTER TABLE Booking ADD CHECK ( BEval IN (0, 1, 2, 3, 4)) ;

ALTER TABLE Booking ADD CONSTRAINT Booking\_PK PRIMARY KEY ( Voyage\_ID, Passenger\_No, Cruise\_ID ) ;

CREATE TABLE Cruise

(

Cruise\_ID VARCHAR2 (8) NOT NULL ,

CName VARCHAR2 (20) NOT NULL ,

CDuration NUMBER (1) NOT NULL

) ;

ALTER TABLE Cruise ADD CHECK ( CDuration IN (2, 3, 4, 5)) ;

ALTER TABLE Cruise ADD CONSTRAINT Cruise\_PK PRIMARY KEY ( Cruise\_ID ) ;

CREATE TABLE Passenger

(

Passenger\_No VARCHAR2 (7) NOT NULL ,

PFName VARCHAR2 (10) NOT NULL ,

PSName VARCHAR2 (10) NOT NULL ,

PAddress VARCHAR2 (20) NOT NULL ,

PGender VARCHAR2 (10) NOT NULL ,

PTelNo NUMBER (12) NOT NULL ,

PEmail VARCHAR2 (30)

) ;

ALTER TABLE Passenger ADD CONSTRAINT Passenger\_PK PRIMARY KEY ( Passenger\_No ) ;

CREATE TABLE Ship

(

S\_No VARCHAR2 (6) NOT NULL ,

SName VARCHAR2 (20) NOT NULL ,

SPort VARCHAR2 (10) NOT NULL ,

STonnage NUMBER (5) NOT NULL

) ;

ALTER TABLE Ship ADD CHECK ( STonnage BETWEEN 25000 AND 60000) ;

ALTER TABLE Ship ADD CONSTRAINT Ship\_PK PRIMARY KEY ( S\_No ) ;

CREATE TABLE Voyage

(

Voyage\_ID VARCHAR2 (7) NOT NULL ,

VName VARCHAR2 (20) NOT NULL ,

VDate DATE NOT NULL ,

VSloc VARCHAR2 (10) NOT NULL ,

VCost NUMBER (4) NOT NULL ,

S\_No VARCHAR2 (6) NOT NULL ,

Cruise\_ID VARCHAR2 (8) NOT NULL

) ;

ALTER TABLE Voyage ADD CHECK ( VSloc IN ('Athens', 'Naples', 'Nice', 'Rome')) ;

ALTER TABLE Voyage ADD CHECK ( VCost BETWEEN 600 AND 5000) ;

ALTER TABLE Voyage ADD CONSTRAINT Voyage\_PK PRIMARY KEY ( Voyage\_ID ) ;

ALTER TABLE Booking ADD CONSTRAINT Booking\_Cruise\_FK FOREIGN KEY ( Cruise\_ID ) REFERENCES Cruise ( Cruise\_ID ) ;

ALTER TABLE Booking ADD CONSTRAINT Booking\_Passenger\_FK FOREIGN KEY ( Passenger\_No ) REFERENCES Passenger ( Passenger\_No ) ;

ALTER TABLE Booking ADD CONSTRAINT Booking\_Voyage\_FK FOREIGN KEY ( Voyage\_ID ) REFERENCES Voyage ( Voyage\_ID ) ;

ALTER TABLE Voyage ADD CONSTRAINT Voyage\_Cruise\_FK FOREIGN KEY ( Cruise\_ID ) REFERENCES Cruise ( Cruise\_ID ) ;

ALTER TABLE Voyage ADD CONSTRAINT Voyage\_Ship\_FK FOREIGN KEY ( S\_No ) REFERENCES Ship ( S\_No ) ;

-- Oracle SQL Developer Data Modeler Summary Report:

--

-- CREATE TABLE 5

-- CREATE INDEX 0

-- ALTER TABLE 16

-- CREATE VIEW 0

-- CREATE PACKAGE 0

-- CREATE PACKAGE BODY 0

-- CREATE PROCEDURE 0

-- CREATE FUNCTION 0

-- CREATE TRIGGER 0

-- ALTER TRIGGER 0

-- CREATE COLLECTION TYPE 0

-- CREATE STRUCTURED TYPE 0

-- CREATE STRUCTURED TYPE BODY 0

-- CREATE CLUSTER 0

-- CREATE CONTEXT 0

-- CREATE DATABASE 0

-- CREATE DIMENSION 0

-- CREATE DIRECTORY 0

-- CREATE DISK GROUP 0

-- CREATE ROLE 0

-- CREATE ROLLBACK SEGMENT 0

-- CREATE SEQUENCE 0

-- CREATE MATERIALIZED VIEW 0

-- CREATE SYNONYM 0

-- CREATE TABLESPACE 0

-- CREATE USER 0

--

-- DROP TABLESPACE 0

-- DROP DATABASE 0

--

-- REDACTION POLICY 0

-- TSDP POLICY 0

--

-- ERRORS 0

-- WARNINGS 0

## Q2 – Populate your database

SQL> select \* from cruise;

**Results:**

CRUISE\_I CNAME CDURATION

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

20000001 Naples 5

20000002 Rome 2

20000003 Nice 3

20000004 Athens 4

20000005 Kefalonia 3

20000006 Cyprus 5

6 rows selected.

SQL> select \* from ship;

**Results:**

S\_NO SNAME SPORT STONNAGE

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

100001 Mary Ann Athens 27000

100002 The Intrepid Rome 43000

100003 Sturgeon Naples 31200

100004 Achilles Nice 59600

100005 Odyssey Rome 34000

100006 Gemini Naples 49500

6 rows selected.

SQL> select \* from voyage;

**Results:**

VOYAGE\_ VNAME VDATE VSLOC VCOST S\_NO CRUISE\_I

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

300001 Rome 1 17-FEB-05 Rome 4500 100002 20000002

300002 Naples 1 29-JAN-05 Naples 900 100003 20000001

300003 Nice 1 31-MAY-04 Nice 1900 100005 20000003

300004 Rome 2 13-FEB-04 Rome 3800 100002 20000002

300005 Athens 1 07-APR-05 Athens 2700 100001 20000004

300006 Naples 2 30-APR-05 Naples 4800 100006 20000001

300007 Athens 2 23-JUL-05 Athens 2100 100001 20000004

300008 Nice 2 10-SEP-05 Nice 4500 100005 20000003

300009 Naples 3 31-AUG-05 Naples 700 100006 20000001

9 rows selected.

SQL> select \* from passenger;

**Results:**

PASSENG PFNAME PSNAME PADDRESS PGENDER PTELNO PEMAIL

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

4000001 Joe Cook 21 Dunhampton Road Male 7658978644 Joe@yahoo.com

4000002 Sam Roads 34 Lingot Drive Male 1562962748

4000003 Ann Kale 7 Magistrate Road Female 7897346736 Ann@gmail.com

4000004 Elizabeth Tiles 19 Lake Road Female 1734566390 Elizabeth@hotmail.com

4000005 John Wick 11 Waterfall Drive Male 7439678378

4000006 Lisa Stone 27 Weavers Wharf Female 7863654674 Lisa@Outlook.com

4000007 Emma Moody 42 Wolverley Road Female 8446789455 Emma@yahoo.com

4000008 Harry Alter 15 Chester Wharf Male 8734595473 Harry@yahoo.com

4000009 Frank Newey 30 Birmingham Drive Male 3868752343

4000010 Maria Smith 3 Churchill Street Female 7438768244 Maria@hotmail.com

10 rows selected.

SQL> select \* from booking;

**Results:**

CRUISE\_I VOYAGE\_ PASSENG BCABNO BCLASS BEVAL

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

20000002 300001 4000001 135 First 3

20000002 300001 4000003 24 First

20000002 300001 4000010 31 Third 0

20000001 300002 4000001 23 Second 2

20000003 300003 4000004 18 Third 1

20000003 300003 4000008 146 First 4

20000003 300003 4000010 45 Third 4

20000002 300004 4000008 44 Third

20000002 300004 4000010 29 Third 3

20000004 300005 4000008 50 Second 2

20000004 300007 4000001 151 First 1

20000004 300007 4000003 49 Second 4

20000003 300008 4000004 48 Second

13 rows selected.

## Q3 – SQL Queries

a. Find the cruise that runs the most number of times

SQL> select v.cruise\_id, count(\*) from voyage v group by cruise\_id;

**Results:**

CRUISE\_I COUNT(\*)

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

20000001 3

20000003 2

20000002 2

20000004 2

b. Find the total number of passengers on all voyages of each cruise

SQL> select b.cruise\_id, count(passenger\_no) from booking b group by cruise\_id;

**Results:**

CRUISE\_I COUNT(PASSENGER\_NO)

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

20000001 1

20000003 4

20000002 5

20000004 3

c. Show the number, name and duration of all cruises along with the total revenue that they each generate

SQL> select v.cruise\_id, cname, cduration, sum(vcost) from voyage v, cruise c where v.cruise\_id(+) = c.cruise\_id group by v.cruise\_id, cname, cduration;

**Results:**

CRUISE\_I CNAME CDURATION SUM(VCOST)

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

Cyprus 5

20000002 Rome 2 8300

20000004 Athens 4 4800

20000001 Naples 5 6400

20000003 Nice 3 6400

Kefalonia 3

6 rows selected.

d. Show details of passenger names and the names of the cruises in which they have participated

SQL> select distinct p.pfname, p.psname, c.cname from passenger p, cruise c, booking b where p.passenger\_no = b.passenger\_no and c.cruise\_id = b.cruise\_id;

**Results:**

PFNAME PSNAME CNAME

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

Joe Cook Athens

Joe Cook Naples

Elizabeth Tiles Nice

Harry Alter Athens

Harry Alter Nice

Joe Cook Rome

Harry Alter Rome

Ann Kale Rome

Maria Smith Rome

Ann Kale Athens

Maria Smith Nice

11 rows selected.

e. Write and test a query to list the name and cost of each cruise. Add a column

that compares the cost of the cruise to the average cost i.e., shows the difference

between the cruise cost and the average cost of all cruises

SQL> select cname, vcost, (select avg(vcost) from voyage) from cruise c, voyage v where c.cruise\_id = v.cruise\_id;

**Results:**

CNAME VCOST (SELECTAVG(VCOST)FROMVOYAGE)

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

Rome 4500 2877.77778

Naples 900 2877.77778

Nice 1900 2877.77778

Rome 3800 2877.77778

Athens 2700 2877.77778

Naples 4800 2877.77778

Athens 2100 2877.77778

Nice 4500 2877.77778

Naples 700 2877.77778

9 rows selected.

f. Find the voyages of cruises that have an occupancy below the average occupancy for voyages of that cruise.

SQL> select voyage\_id, count(passenger\_no), (select avg(count(passenger\_no)) from booking group by voyage\_id) as avg from booking group by voyage\_id;

**Results:**

VOYAGE\_ COUNT(PASSENGER\_NO) AVG

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

300001 3 1.85714286

300002 1 1.85714286

300003 3 1.85714286

300004 2 1.85714286

300005 1 1.85714286

300007 2 1.85714286

300008 1 1.85714286

7 rows selected.

g. Find passengers evaluation of each voyage of each cruise they have taken.

SQL> select cruise\_id, voyage\_id, passenger\_no, beval from booking;

**Results:**

CRUISE\_I VOYAGE\_ PASSENG BEVAL

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

20000002 300001 4000001 3

20000002 300001 4000003

20000002 300001 4000010 0

20000001 300002 4000001 2

20000003 300003 4000004 1

20000003 300003 4000008 4

20000003 300003 4000010 4

20000002 300004 4000008

20000002 300004 4000010 3

20000004 300005 4000008 2

20000004 300007 4000001 1

20000004 300007 4000003 4

20000003 300008 4000004

13 rows selected.

h. Find the name, duration and cost of a cruise as well as the voyage date, starting location and the name of the ship.

SQL> select cname, cduration, vcost, vdate, vsloc, sname from cruise c, voyage v, ship s where c.cruise\_id = v.cruise\_id and s.s\_no = v.s\_no;

**Results:**

CNAME CDURATION VCOST VDATE VSLOC SNAME

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

Rome 2 4500 17-FEB-05 Rome The Intrepid

Naples 5 900 29-JAN-05 Naples Sturgeon

Nice 3 1900 31-MAY-04 Nice Odyssey

Rome 2 3800 13-FEB-04 Rome The Intrepid

Athens 4 2700 07-APR-05 Athens Mary Ann

Naples 5 4800 30-APR-05 Naples Gemini

Athens 4 2100 23-JUL-05 Athens Mary Ann

Nice 3 4500 10-SEP-05 Nice Odyssey

Naples 5 700 31-AUG-05 Naples Gemini

9 rows selected.

i. Find the cruise id, voyage id, passenger no, the passsengers cabin number, class, their evaluation as well as their name and contact details.

SQL> select cruise\_id, voyage\_id, b.passenger\_no, bcabno, bclass, beval, pfname, psname, ptelno, pemail from booking b,

passenger p where b.passenger\_no = p.passenger\_no;

**Results:**

CRUISE\_I VOYAGE\_ PASSENG BCABNO BCLASS BEVAL PFNAME PSNAME PTELNO PEMAIL

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

20000002 300001 4000001 135 First 3 Joe Cook 7658978644 Joe@yahoo.com

20000002 300001 4000003 24 First Ann Kale 7897346736 Ann@gmail.com

20000002 300001 4000010 31 Third 0 Maria Smith 7438768244 Maria@hotmail.com

20000001 300002 4000001 23 Second 2 Joe Cook 7658978644 Joe@yahoo.com

20000003 300003 4000004 18 Third 1 Elizabeth Tiles 1734566390 Elizabeth@hotmail.com

20000003 300003 4000008 146 First 4 Harry Alter 8734595473 Harry@yahoo.com

20000003 300003 4000010 45 Third 4 Maria Smith 7438768244 Maria@hotmail.com

20000002 300004 4000008 44 Third Harry Alter 8734595473 Harry@yahoo.com

20000002 300004 4000010 29 Third 3 Maria Smith 7438768244 Maria@hotmail.com

20000004 300005 4000008 50 Second 2 Harry Alter 8734595473 Harry@yahoo.com

20000004 300007 4000001 151 First 1 Joe Cook 7658978644 Joe@yahoo.com

20000004 300007 4000003 49 Second 4 Ann Kale 7897346736 Ann@gmail.com

20000003 300008 4000004 48 Second Elizabeth Tiles 1734566390 Elizabeth@hotmail.com

13 rows selected.

j. Show passengers evaluation of each voyage of each cruise they have taken that is a 3 or higher as well as their details, cabin number and class.

SQL> select b.cruise\_id, b.voyage\_id, b.passenger\_no, pfname, psname, paddress, ptelno, bcabno, bclass, beval from booking b, passenger p where b.passenger\_no = p.passenger\_no and beval >= 3;

**Results:**

CRUISE\_I VOYAGE\_ PASSENG PFNAME PSNAME PADDRESS PTELNO BCABNO BCLASS BEVAL

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

20000002 300001 4000001 Joe Cook 21 Dunhampton Road 7658978644 135 First 3

20000003 300003 4000008 Harry Alter 15 Chester Wharf 8734595473 146 First 4

20000003 300003 4000010 Maria Smith 3 Churchill Street 7438768244 45 Third 4

20000002 300004 4000010 Maria Smith 3 Churchill Street 7438768244 29 Third 3

20000004 300007 4000003 Ann Kale 7 Magistrate Road 7897346736 49 Second 4

5 rows selected.

## Q4 – View

**A)**

SQL> desc cruise\_view;

Name Null? Type

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

CRUISE\_ID NOT NULL VARCHAR2(8)

CNAME NOT NULL VARCHAR2(20)

CDURATION NOT NULL NUMBER(1)

VOYAGE\_ID NOT NULL VARCHAR2(7)

VNAME NOT NULL VARCHAR2(20)

VDATE NOT NULL DATE

VSLOC NOT NULL VARCHAR2(10)

VCOST NOT NULL NUMBER(4)

SQL> select \* from cruise\_view;

**Results:**

CRUISE\_I CNAME CDURATION VOYAGE\_ VNAME VDATE VSLOC VCOST

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

20000002 Rome 2 300001 Rome 1 17-FEB-05 Rome 4500

20000001 Naples 5 300002 Naples 1 29-JAN-05 Naples 900

20000003 Nice 3 300003 Nice 1 31-MAY-04 Nice 1900

20000002 Rome 2 300004 Rome 2 13-FEB-04 Rome 3800

20000004 Athens 4 300005 Athens 1 07-APR-05 Athens 2700

20000001 Naples 5 300006 Naples 2 30-APR-05 Naples 4800

20000004 Athens 4 300007 Athens 2 23-JUL-05 Athens 2100

20000003 Nice 3 300008 Nice 2 10-SEP-05 Nice 4500

20000001 Naples 5 300009 Naples 3 31-AUG-05 Naples 700

9 rows selected.

**Bi)**

SQL> update cruise\_view set cruise\_Id = 20000007 where cruise\_id = 20000001;

**Results:**

update cruise\_view set cruise\_Id = 20000007 where cruise\_id = 20000001

\*

ERROR at line 1:

ORA-01779: cannot modify a column which maps to a non key-preserved table

This is because the view is a join-view and therefore requires the column to come from a key preserved table which is not the case with this implementation.

SQL> update cruise\_view set cname = 'test' where cruise\_id = 20000001;

**Results:**

update cruise\_view set cname = 'test' where cruise\_id = 20000001

\*

ERROR at line 1:

ORA-01779: cannot modify a column which maps to a non key-preserved table

This is because the view is a join-view and therefore requires the column to come from a key preserved table which is not the case with this implementation.

SQL> update cruise\_view set cduration = 0 where cruise\_id = 20000001;

**Results:**

update cruise\_view set cduration = 0 where cruise\_id = 20000001

\*

ERROR at line 1:

ORA-01779: cannot modify a column which maps to a non key-preserved table

This is because the view is a join-view and therefore requires the column to come from a key preserved table which is not the case with this implementation.

SQL> update cruise\_view set voyage\_id = 300002 where cruise\_id = 20000001;

**Results:**

update cruise\_view set voyage\_id = 300002 where cruise\_id = 20000001

\*

ERROR at line 1:

ORA-00001: unique constraint (OPS$1007022.VOYAGE\_PK) violated

This is because the column is a primary key and therefore its rows must be unique.

SQL> update cruise\_view set vname = 'test' where cruise\_id = 20000001;

**Results:**

3 rows updated.

This is because the view is a join-view and therefore requires the column to come from a key preserved table which is the case with this implementation.

SQL> update cruise\_view set vdate = '30-JAN-2005' where cruise\_id = 20000001;

**Results:**

3 rows updated.

This is because the view is a join-view and therefore requires the column to come from a key preserved table which is the case with this implementation.

SQL> update cruise\_view set vsloc = 'Athens' where cruise\_id = 20000001;

**Results:**

3 rows updated.

This is because the view is a join-view and therefore requires the column to come from a key preserved table which is the case with this implementation.

SQL> update cruise\_view set vcost = 600 where cruise\_id = 20000001;

**Results:**

3 rows updated.

This is because the view is a join-view and therefore requires the column to come from a key preserved table which is the case with this implementation.

**Bii)**

SQL> insert into cruise\_view

2 values(20000001, 'Athens', 2, 300001, 'test', '30-jan-2015', 'Athens', 600);

**Results:**

insert into cruise\_view

\*

ERROR at line 1:

ORA-01779: cannot modify a column which maps to a non key-preserved table

This is because the view is a join-view and therefore requires the column to come from a key preserved table which is not the case with this implementation.

SQL> insert into cruise\_view

2 values(20000002, 'Nice', 3, 300002, 'test', '30-jan-2015', 'Nice', 700);

**Results:**

insert into cruise\_view

\*

ERROR at line 1:

ORA-01779: cannot modify a column which maps to a non key-preserved table

This is because the view is a join-view and therefore requires the column to come from a key preserved table which is not the case with this implementation.

**Biii)**

SQL> delete from cruise\_view where voyage\_id = 300001;

**Results:**

delete from cruise\_view where voyage\_id = 300001

\*

ERROR at line 1:

ORA-02292: integrity constraint (OPS$1007022.BOOKING\_VOYAGE\_FK) violated - child record found

This is because of the child records stored in the booking table which use this particular voyage\_id.

## Q5 – Triggers

**a)**

SQL> create or replace trigger cruise\_duration

2 after insert on voyage

3 FOR EACH ROW

4 when (NEW.cruise\_id = 20000001)

5 begin

6 if (:NEW.vdate like '%NOV%')

7 then raise\_application\_error(-20100,

8 'Cannot add a voyage of 5 days duration

9 in november');

10 end if;

11 end;

12 /

**Results:**

Trigger created.

SQL> insert into voyage values

2 (1111111, 'test', '01-NOV-05', 'test', 100002, 20000001);

**Results:**

insert into voyage values

\*

ERROR at line 1:

ORA-20100: Cannot add a voyage of 5 days duration

in november

ORA-06512: at "OPS$1007022.CRUISE\_DURATION", line 3

ORA-04088: error during execution of trigger 'OPS$1007022.CRUISE\_DURATION'

**b)**

SQL> create or replace trigger pass\_eval

2 after insert or update of peval on booking

3 for each row

4 begin

5 if (:NEW.peval = 0) then

6 insert into pass\_audit

7 values (:OLD.pfname, :OLD.psname, :NEW.cruise\_id, :OLD.vdate, :OLD.s\_no, :NEW.peval);

8 end if;

9 end;

10 /

**Results:**

Warning: Trigger created with compilation errors.

SQL> show errors

Errors for TRIGGER PASS\_EVAL:

LINE/COL ERROR

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

4/9 PLS-00049: bad bind variable 'OLD.PFNAME'

4/22 PLS-00049: bad bind variable 'OLD.PSNAME'

4/51 PLS-00049: bad bind variable 'OLD.VDATE'

4/63 PLS-00049: bad bind variable 'OLD.S\_NO'

**c)**

SQL> create or replace trigger voyage\_date

2 before delete on voyage

3 for each row

4 begin

5 if (:OLD.vdate < sysdate) then

6 raise\_application\_error(-20101,

7 'Cannot delete a voyage that has

8 already taken place');

9 end if;

10 end;

11 /

**Results:**

Trigger created.

SQL> delete from voyage where voyage\_id = 300009;

**Results:**

delete from voyage where voyage\_id = 300009

\*

ERROR at line 1:

ORA-20101: Cannot delete a voyage that has

already taken place

ORA-06512: at "OPS$1007022.VOYAGE\_DATE", line 3

ORA-04088: error during execution of trigger 'OPS$1007022.VOYAGE\_DATE'

## Q6 – Report

Data warehousing is a way of storing historical data possibly five to ten years old from of what was at the time current operational data which came from the front line operation data storage, due to this the data stored is only accurate as of the time of insertion because over time the structure of operational data changes to meet the functional requirements of the companies clients, it is generally a relational database and the data does not get updated therefore the data warehouse can be seen as version control over a long period of time in order to keep a record of how the data's structure changed over time.

A data warehouse is useful because it provides analysts with a large amount of data in order to discover trends that could give the company an edge over its competitors or simply a boost in sales. However initially when the data warehouse is set up the data may not prove useful or it may not be possible to make sense of the data, and as structures change it may be hard to allow for such dynamic data structure therefore data warehousing is useful if set up properly but it is not without its issues which is backed up by the work of **(Alhyasat, E. B., & Al-Dalahmeh, M. 2013) who states** *"The high risk/high return management of Data Warehouses…is a complex undertaking, since the challenge of the phenomenal growth of high volumes of data can be risky in terms of cost effectiveness and security manners".* This is also backed up by the work of **(Al-Debei, M. M. 2011)** who describes some of the issues of data warehousing such as the design of the data warehouse you go for affecting how well it will function, as well as how well the data can be inserted, how much that can extend to, the quality and security issues.

References -

1) Golfarelli, M., & Rizzi, S. (2011). Data warehouse testing: A prototype-based methodology. Information and Software Technology, 53(11), 1183–1198.

2) Alhyasat, E. B., & Al-Dalahmeh, M. (2013). Data Warehouse Success and Strategic Oriented Business Intelligence: A Theoretical Framework. Journal of Management Research, 5(3), 169–184.

3) De Mul, M., Alons, P., van der Velde, P., Konings, I., Bakker, J., & Hazelzet, J. (2012). Development of a clinical data warehouse from an intensive care clinical information system. Computer Methods and Programs in Biomedicine, 105(1), 22–30.

4) Al-Debei, M. M. (2011). Data Warehouse as a Backbone for Business Intelligence: Issues and Challenges. European Journal of Economics, Finance & Administrative Sciences, (33), 153–166. Retrieved from http://ezproxy.lib.swin.edu.au/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=bth&AN=67737744&site=ehost-live&scope=site