ADDB Assignment 1

|  |  |
| --- | --- |
| Student Name: | Karl Dicks |
| Student Number: | 17667327 |
| Course: | BCAD2 |
| Subject: | ADDB7311 |
| Lecturer: | Nirasha Ramckurran |
| Assignment: | Assignment 1 |
| Due Date: | 02/05/2019 |

# Question 1:

There are five main types of database models, including the hierarchical, entity-relationship, relational, network, and object-oriented models. The most commonly used model is the relational model, which allows relevant data to be stored in tables with relationships connecting these tables together. Data models define how the logical structure of the database is modelled. They allow for abstraction – which hides irrelevant details from the users of the system. Models also describe how the data in a system is connected to each other, processed, and stored. (Singh, 2015)

The five types of database models are described below:

**Hierarchical Model:**

The hierarchical model represents data as a hierarchical tree structure. Each branch of the hierarchy represents a number of related records. It uses a one to many, or one to one relationship between each branch (parent) and multiple related records (children). Each related record (child) will have only one parent – it cannot have more than one. Hierarchical models are not flexible, and were mainly used with mainframe computers. They are an old database modelling technique, and have been replaced by newer (more flexible) models, such as the relational model. (Prabhjot, 2017)

The hierarchical model is implemented when there is a concrete hierarchy, such as in a business, where there are levels such as the board of directors, CEO, CFO, etc. It describes the functional levels, and the relationships between them. There would be fewer relationships in a hierarchical model, as each child node has only one relationship that is between the parent and itself. It cannot have a relationship between other child records or parent records. (Prabhjot, 2017)

Some factors to take into consideration when implementing the database would be the following:

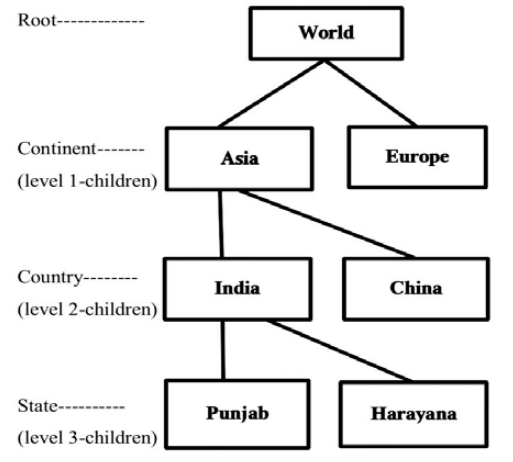
Disadvantages:

1. The hierarchical model is not flexible – it cannot adapt easily to changes to the database, such as new parent records, etc.
2. Because of the reduced number of relationships, data retrieval will be slower than other models.
3. Implementation complexity is increased with this type of database model. Requires knowledge of physical data storage characteristics.
4. There is a lack of structural independence – the reason why changes cannot be made as easily as in the relational model. Requires changes to applications. (Singh, 2015)
5. There is a lack of standards.
6. There is no data definition or data manipulation language in the DBMS. (Carlos Coronel, 2017)

Advantages:

1. Parent/ child relationships promotes conceptual simplicity.
2. This model is efficient, with one to many relationships.
3. Parent/ child relationships promote data integrity.
4. Promotes data sharing.
5. Database security is provided and enforced by DBMS. (Carlos Coronel, 2017)

Example of the hierarchical database model:



Description of example:

The model starts with the root data – being the World record, and then expands to include Asia and Europe child records. There are therefore relationships between Asia and World, and Europe and World. Asia contains multiple countries including India and China. Within India there are multiple states, two of which are Punjab and Harayana.

Relationships include the following:

1. World has one to many relationship between continents. There is one parent (World), and multiple children (Asia, Europe).
2. Each continent can have multiple countries (Asian continent can have India, and China countries).
3. Each country can have multiple states, regions, oblasts, or provinces. In the case of India, there are multiple states, including Punjab and Harayana.

**Entity Relationship Model:**

Developed by Peter Chen in 1976. (Eid, 2012)

This model is based on real world entities and relationships between them. An entity is a “thing”, such as a car, building, etc. This will be stored as a database object in a real world database. An entity can contain many attributes that describe its properties. For example, attributes about a car could be model, age, and VIN number. These attributes describe the instance of the vehicle. Relationships are the logical associations among entities. There are different types of relationship cardinalities including the following: (Eid, 2012)

1. One to One
2. One to Many
3. Many to One
4. Many to Many

(Eid, 2012)

A normalized database should only have one to many or one to one relationships. Many to many relationships will cause redundant data, and therefore can lead to update, insert, and delete errors when using the database. “This model is a high level description of the data and the relationships among entities. It is a nontechnical method that provides a standard and logical way of visualizing the data and free from ambiguous.” (Prabhjot, 2017)

Some factors to take into consideration when implementing the database would be the following:

1. No two entities should be identical – they should hold unique data.
2. The entities should contain unique keys that identify an entity in an entity set.
3. Should conform to constraints such as mapping cardinality, and participation ratio.
   1. Mapping cardinality: describes the number of entities associated to other entities.
   2. Participation ratio: describes whether there is partial or total participation of one entity to another. (Prabhjot, 2017)

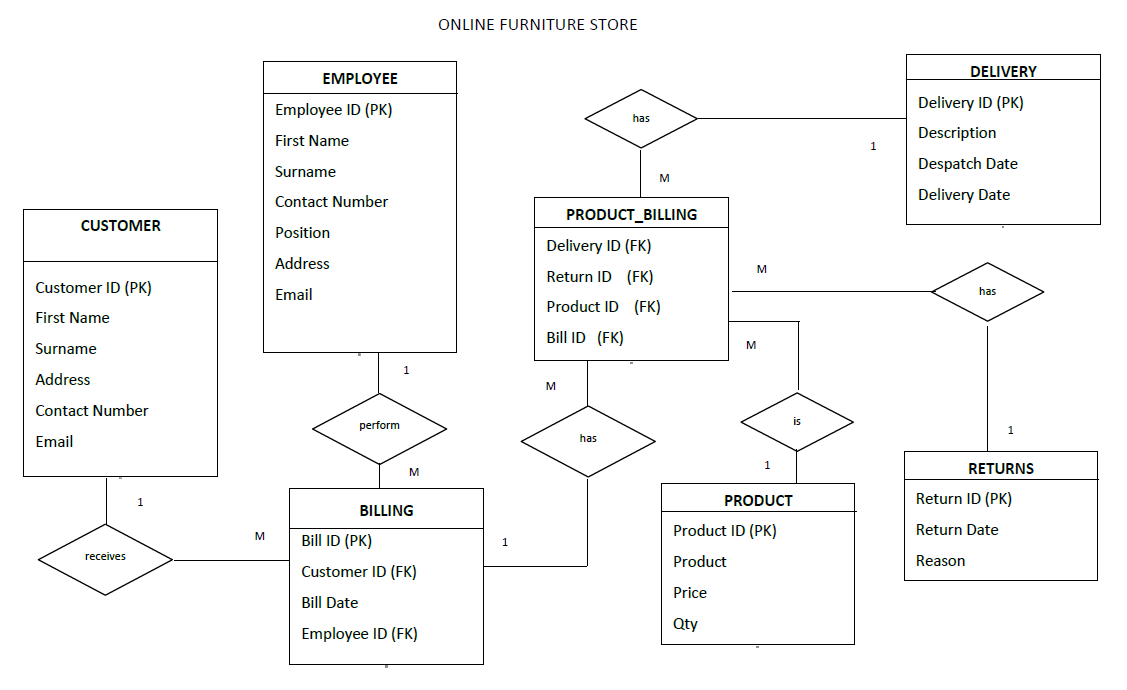
Disadvantages:

1. Limited constraint representation.
2. Limited relationship representation.
3. There is no data manipulation language.
4. Loss of information content occurs when attributes are removed from entities to avoid crowded displays. (Carlos Coronel, 2017)

Advantages:

1. It is integrated with the dominant relational model.
2. Visual modelling yields exceptional conceptual simplicity.
3. Visual representation makes it an effective communication tool. (Carlos Coronel, 2017)

Example of Entity Relationship Model:



(The Independent Institute of Education, 2019)

Description of example:

This model has no “start” and “end”. The entities Customer, Employee, Billing, Product\_Billing, Product, Returns, and Delivery show entities that are described by their attributes (properties). Relationship names describe the relationships between entities, such as a customer “receives” as billing. This improves the readability and ease of understanding of the model. (The Independent Institute of Education, 2019)

Relationships include the following:

1. One customer can receive many billings.
2. One employee can perform many billings.
3. One billing can have many product billings.
4. One delivery can have many product billings.
5. Many product billings can have a return.
6. A product can be in many product billings.

(The Independent Institute of Education, 2019)

**Relational Model:**

Developed by E.F Codd in 1970. (Radoslava Kraleva, 2018)

This model is the most widely used database model – essentially the only model used around the world. In this model, data is stored in two dimensional tables, also referred to as relations. Attributes describe each relation (table) instance, including its properties that are stored in columns in the table. Tables have a definite number of columns, but can have an infinite number of rows (tuples). For example, there are only a certain number of attributes of a car, but there can be many cars. (Joseph V. Homan, 2009)

Attribute domain is all possible values of the attribute. For example, it may have to be an integer between the range of 1 and 100, or a string using the English/ Latin alphabet. Each table is called a relation, with attributes (that describe the column), columns (that hold like values – Names, etc.), and tuples (which are rows of values that describe an entity instance – describe a “thing” such as a car). Each tuple (row) in a relation contains a unique value that identifies it within the relation, and each column in a relation contains values from the same domain. A set of tuples is called a record, and hence a relational model is often called a Record-Based Model. Columns of a table are called attributes (the columns store attributes of the relation). (Singh, 2015)

Relations in a real world database should be normalized, which eliminates data redundancy, and has benefits of reducing update, delete, and insert errors when working with the database. (Radoslava Kraleva, 2018)

Some factors to take into consideration when implementing the database would be the following:

1. The database must be normalized, in order to improve performance and decrease data redundancy – which can cause errors when working with the data stored in the database.
2. Keys (primary key) must be used to uniquely identify each tuple in a relation.
3. Primary key must be unique, non-changing and not null. This promotes entity integrity constraints in the database system.
4. Foreign keys must be either null or reference primary keys of another table. This promotes referential integrity constraints.
5. The business should implement this type of model, as it is the most widely used – and for good reason. Scalability, data integrity, performance, and the ability to easily understand the model are just some of its benefits.
6. There should be no duplicate relations in the database. (Prabhjot, 2017)

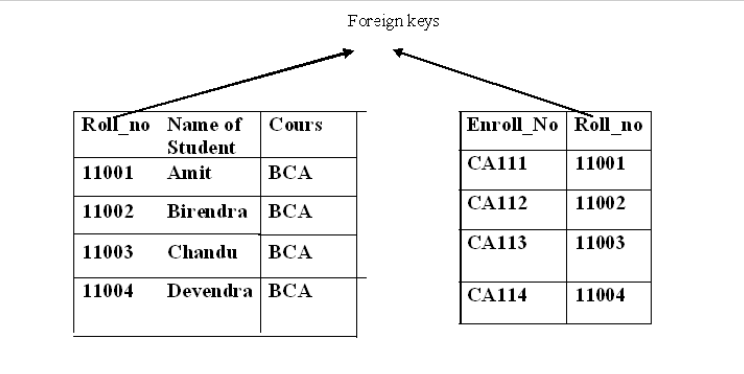
Disadvantages:

1. The RDBMS requires substantial hardware and system software overhead.
2. Conceptual simplicity gives relatively untrained people the tools to use a good system poorly, and if unchecked, it may produce the same data anomalies found in file systems.
3. It may promote islands of information problems as individuals and departments can easily develop their own applications. (Carlos Coronel, 2017)

Advantages:

1. Powerful RDBMS isolates the end user from physical level details and improves implementation and management simplicity.
2. Ad hoc query capability is based on SQL.
3. Tabular view substantially improves conceptual simplicity, thereby promoting easier database design, implementation, management and use.
4. Structural independence is promoted by the use of independent tables. Changes in a tables structure do not affect data access or application programs. (Carlos Coronel, 2017)

Example of Relational Model:



(Singh, 2015)

Description of example:

This relational database contains data about students, enrolment, and courses. Roll relation contains roll\_id, name, and course attributes. Enrol relation contains enrol\_id, roll\_no. These attributes describe each of the relations, and describe what data each of them hold.

Relationships include the following:

1. One student may enrol in multiple courses.
2. Each course may contain multiple students.
3. Foreign key Roll\_no in Enrol table can either be null or reference PK of Roll relation.

(Singh, 2015)

**Network Model:**

This model was the most used database model until the Relational Model was introduced. It is an extension to the Hierarchical Model. It looks much like a graph, featuring many to many relationships as well as one to one relationships. Accessing data in the network model is much simpler when compared with the hierarchical model. There is always a connection (relationship) between the parent and child segments because it depends on the parent-child relationship. “Data In network data model is organized as a collection of graphs of record that are related with pointers. A pointer is a physical address which identifies where the next record can be found on the disk. “ (Singh, 2015)

Data independence is also improved in the network model when compared to the hierarchical model. There are however certain drawbacks of the network model. (Singh, 2015)

Some factors to take into consideration when implementing the database would be the following:

1. This model is not widely adopted anymore, since the development of the relational model.
2. There are some significant drawbacks in using this model, such as scalability issues, and complexity.
3. Many to many relationships cause database complexity and data redundancy.
4. Insertion, deletion, and updates are more complex due to the requirement of pointers.

(Singh, 2015)

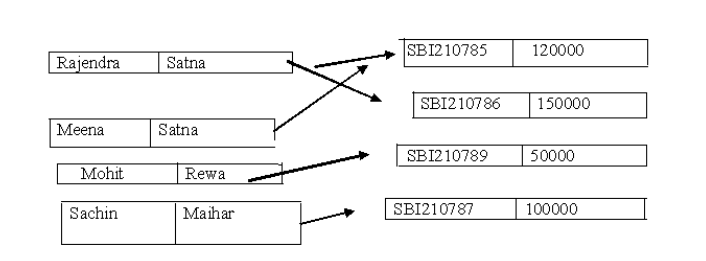
Disadvantages:

1. Structural complexity limits efficiency. It is still a navigational system.
2. Navigational system yields complex implementation, application development, and management.
3. Structural changes require changes in all application programs. (Carlos Coronel, 2017)

Advantages:

1. Conceptual simplicity is at least equal to that of the hierarchical model.
2. It handles more relationship types, such as many to many, and multipart.
3. Data owner/ member relationship promotes data integrity.
4. There is conformance to standards.
5. It includes data definition language (DDL) and data manipulation language (DML) in DBMS. (Carlos Coronel, 2017)

Example of Network Model:



(Singh, 2015)

Description of example:

This network model shows a loan database, which includes its customers and types of loans. Customer details include first name, and surname. Types of loans include loan ID, and loan amount.

Relationships include the following:

1. A customer can take out multiple different loans.
2. A loan type can have many different customers.

This is a many to many relationship. (Singh, 2015)

**Object-Oriented Model:**

This model uses objects, which include relationships and their data. The OODM is the basis for the object-oriented database management system. It is similar to the relational model, in that the object is described by its factual content. However, unlike the relational model, it contains information about the relationships between the facts/data within the object and between other objects in the database. The facts within in the object are therefore given greater meaning. OODM is known as a semantic model, as semantic indicates meaning. (Carlos Coronel, 2017)

An object is an abstraction of a real world entity, similar to a relational entity. An object however only represents one individual occurrence of an entity. Attributes describe an object, much like attributes in the relational model. For example, a student may have an ID, name, surname, etc. Objects that share similar characteristics are grouped together into classes. Classes are organized into a class hierarchy, which represents an upside down tree. Each class has only one parent, for example a customer and employee will share person class. This is similar to the hierarchy structure. (Carlos Coronel, 2017)

This model promotes object inheritance, which allows one object to inherit attributes from another object. For example, a customer and employee will inherit attributes and methods from the person object. (Carlos Coronel, 2017)

Some factors to take into consideration when implementing the database would be the following:

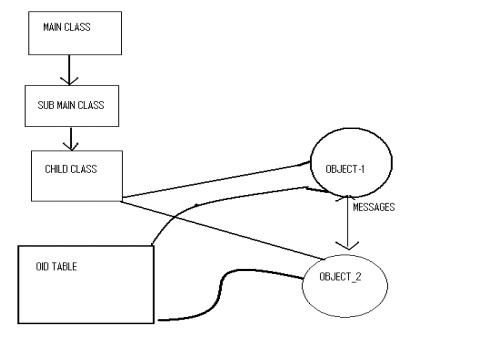
Disadvantages:

1. Slow development of standards, which caused vendors to supply their own improvements, thus eliminating a widely accepted standard.
2. There is a steep learning curve.
3. It is a complex navigational system.
4. There is a high system overhead, which slows transactions. (Carlos Coronel, 2017)

Advantages:

1. Inheritance promotes data integrity.
2. Semantic content is included.
3. Visual representation includes semantic content. (Carlos Coronel, 2017)

Example of Object-Oriented Model:



(Singh, 2015)

Description of example:

This object-oriented model example shows a main class, which has a subclass named “sub main class”. This subclass has a child class, and the child class contains two objects. Object 1 and Object 2 have a relationship, which allows them to retrieve information from each other. Both Object 1 and Object 2 are able to retrieve information from the OID table.

Relationships include the following:

1. Main class has one or many sub classes.
2. Sub main class has one or many child classes.
3. Child class can contain one or many objects.

(Singh, 2015)

# Question 2:

Line 1: Incorrect syntax. Replace “set server output on” with “set serveroutput on;”

Line 2: Incorrect syntax. Replace “declaring” with “declare”

Line 3: Incorrect variable name. Replace variable “cust name” with “cust\_name”

Line 3: Variable data type length not set. Replace “varchar2” with “varchar2(50)”

Line 4: Incorrect variable name. Replace “b date” with “b\_date”

Line 12: Incorrect/ invalid alias used. Replace “bill.staff\_id” with “b.staff\_id”

Line 16: Missing semicolon. Replace “and b.bill\_date = '10 November 2016'” with “and b.bill\_date = '10 November 2016';”

Line 20: Incorrect variable name used, and semicolon missing. Replace “dbms\_output.put\_line('VEHICLE: ' || vehicle)” with “dbms\_output.put\_line('VEHICLE: ' || veh);”

# Question 3:

CREATE TABLE CUSTOMER (

CUSTOMER\_ID INT PRIMARY KEY NOT NULL,

FIRST\_NAME VARCHAR(25) NOT NULL,

SURNAME VARCHAR(25) NOT NULL,

ADDRESS VARCHAR(50) NOT NULL,

CONTACT\_NUMBER VARCHAR(25) NOT NULL,

EMAIL VARCHAR(25) NOT NULL

);

/

CREATE TABLE EMPLOYEE (

EMPLOYEE\_ID CHAR(6) PRIMARY KEY NOT NULL,

FIRST\_NAME VARCHAR(25) NOT NULL,

SURNAME VARCHAR(25) NOT NULL,

CONTACT\_NUMBER VARCHAR(25) NOT NULL,

POSITION VARCHAR(25),

ADDRESS VARCHAR(50) NOT NULL,

EMAIL VARCHAR(25) NOT NULL

);

/

CREATE TABLE DELIVERY (

DELIVERY\_ID INT PRIMARY KEY NOT NULL,

DESCRIPTION VARCHAR(250) NOT NULL,

DISPATCH\_DATE DATE,

DELIVERY\_DATE DATE

);

/

CREATE TABLE RETURNS (

RETURN\_ID CHAR(6) PRIMARY KEY NOT NULL,

RETURN\_DATE DATE NOT NULL,

REASON VARCHAR(250)

);

/

CREATE TABLE PRODUCT (

PRODUCT\_ID INT PRIMARY KEY NOT NULL,

PRODUCT VARCHAR(100) NOT NULL,

PRICE DECIMAL(9, 2) NOT NULL,

QTY INT NOT NULL

);

CREATE TABLE BILLING (

BILL\_ID INT PRIMARY KEY NOT NULL,

CUSTOMER\_ID INT,

BILL\_DATE DATE NOT NULL,

EMPLOYEE\_ID CHAR(6),

CONSTRAINT FK\_CUSTOMER\_ID FOREIGN KEY (CUSTOMER\_ID) REFERENCES CUSTOMER(CUSTOMER\_ID),

CONSTRAINT FK\_EMPLOYEE\_ID FOREIGN KEY (EMPLOYEE\_ID) REFERENCES EMPLOYEE(EMPLOYEE\_ID)

);

/

CREATE TABLE PRODUCT\_BILLING (

DELIVERY\_ID INT,

RETURN\_ID CHAR(6),

PRODUCT\_ID INT,

BILL\_ID INT,

CONSTRAINT FK\_DELIVERY\_ID FOREIGN KEY (DELIVERY\_ID) REFERENCES DELIVERY(DELIVERY\_ID),

CONSTRAINT FK\_RETURN\_ID FOREIGN KEY (RETURN\_ID) REFERENCES RETURNS(RETURN\_ID),

CONSTRAINT FK\_PRODUCT\_ID FOREIGN KEY (PRODUCT\_ID) REFERENCES PRODUCT(PRODUCT\_ID),

CONSTRAINT FK\_BILL\_ID FOREIGN KEY (BILL\_ID) REFERENCES BILLING(BILL\_ID)

);

/

INSERT ALL

INTO CUSTOMER VALUES (11011, 'Jeffery', 'Smith', '18 Water rd', '0877277521', 'jef@isat.com')

INTO CUSTOMER VALUES (11012, 'Alex', 'Hendricks', '22 Water rd', '0863257857', 'ah@mcom.co.za')

INTO CUSTOMER VALUES (11013, 'Johnson', 'Clark', '101 Summer lane', '0834567891', 'jclark@mcom.co.za')

INTO CUSTOMER VALUES (11014, 'Henry', 'Jones', '55 Mountain way', '0612547895', 'hj@isat.co.za')

INTO CUSTOMER VALUES (11015, 'Andre', 'Williams', '5 Main rd', '0827238521', 'aw@mcal.co.za')

SELECT \* FROM dual;

/

INSERT ALL

INTO EMPLOYEE VALUES ('emp101', 'Roan', 'Davis', '0877277521', 'sales', '10 main Road', 'rd@isat.com')

INTO EMPLOYEE VALUES ('emp102', 'Billy', 'Marks', '0837377522', 'marketing', '18 water road', 'bmark@isat.com')

INTO EMPLOYEE VALUES ('emp103', 'Chadwin', 'Andrews', '0817117523', 'sales', '21 circle lane', 'ca@isat.com')

INTO EMPLOYEE VALUES ('emp104', 'Wayne', 'Dryer', '0797215244', 'sales', '1 sea road', 'dryer@isat.com')

INTO EMPLOYEE VALUES ('emp105', 'Jaci', 'Samson', '0827122255', 'manager', '12 main road', 'samjax@isat.com')

SELECT \* FROM dual;

/

INSERT ALL

INTO DELIVERY VALUES (511, 'Delivery contains glass items - fragile', '10 May 2017', '15 May 2017')

INTO DELIVERY VALUES (512, 'Delivery of wooden items', '12 May 2017', '15 May 2017')

INTO DELIVERY VALUES (513, 'No description available', '12 May 2017', '17 May 2017')

INTO DELIVERY VALUES (514, 'Delivery contains glass items - fragile', '12 May 2017', '15 May 2017')

INTO DELIVERY VALUES (515, 'Delivery contains glass items - fragile', '18 May 2017', '19 May 2017')

INTO DELIVERY VALUES (516, 'No description available', '20 May 2017', '25 May 2017')

INTO DELIVERY VALUES (517, 'Delivery of wooden items', '25 May 2017', '27 May 2017')

SELECT \* FROM dual;

/

INSERT ALL

INTO RETURNS VALUES ('ret001', '25 May 2017', 'Customer not satisfied with product')

INTO RETURNS VALUES ('ret002', '25 May 2017', 'Product missing part')

SELECT \* FROM dual;

/

INSERT ALL

INTO PRODUCT VALUES (7111, 'Four Piece Wall Unit', 10999, 10)

INTO PRODUCT VALUES (7112, 'Plasma Stand Unit', 7999, 8)

INTO PRODUCT VALUES (7113, 'Leather Recliner', 5999, 8)

INTO PRODUCT VALUES (7114, 'Leather Lazy Boy', 7999, 5)

INTO PRODUCT VALUES (7115, '6 Piece Fabric Suite', 17999, 15)

INTO PRODUCT VALUES (7116, '6 Piece Leather Suite', 27999, 12)

INTO PRODUCT VALUES (7117, '6 Seater Oak Dining table', 11999, 3)

SELECT \* FROM dual;

/

INSERT ALL

INTO BILLING VALUES (8111, 11011, '15 May 2017', 'emp103')

INTO BILLING VALUES (8112, 11013, '15 May 2017', 'emp101')

INTO BILLING VALUES (8113, 11012, '17 May 2017', 'emp101')

INTO BILLING VALUES (8114, 11015, '17 May 2017', 'emp102')

INTO BILLING VALUES (8115, 11011, '17 May 2017', 'emp102')

INTO BILLING VALUES (8116, 11015, '18 May 2017', 'emp103')

INTO BILLING VALUES (8117, 11012, '19 May 2017', 'emp101')

INTO BILLING VALUES (8118, 11013, '19 May 2017', 'emp105')

SELECT \* FROM dual;

/

INSERT ALL

INTO PRODUCT\_BILLING VALUES (512, null, 7113, 8115)

INTO PRODUCT\_BILLING VALUES (511, null, 7111, 8111)

INTO PRODUCT\_BILLING VALUES (512, null, 7111, 8114)

INTO PRODUCT\_BILLING VALUES (514, 'ret001', 7113, 8113)

INTO PRODUCT\_BILLING VALUES (516, null, 7115, 8112)

INTO PRODUCT\_BILLING VALUES (515, 'ret002', 7114, 8113)

INTO PRODUCT\_BILLING VALUES (517, null, 7113, 8115)

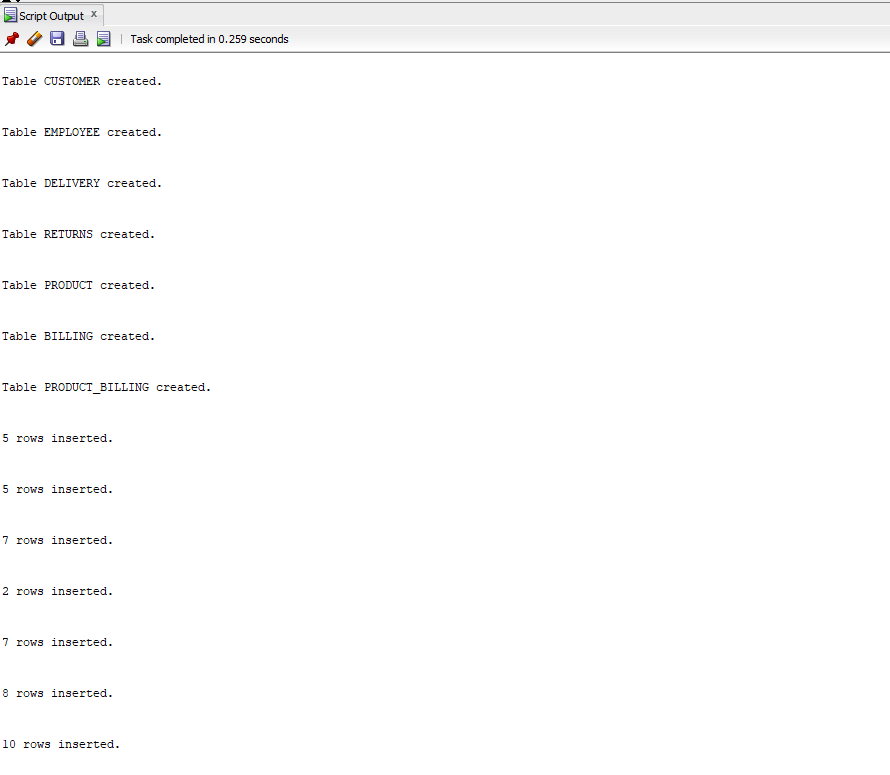
INTO PRODUCT\_BILLING VALUES (511, null, 7112, 8118)

INTO PRODUCT\_BILLING VALUES (513, null, 7111, 8117)

INTO PRODUCT\_BILLING VALUES (512, null, 7115, 8116)

SELECT \* FROM dual;

/



# Question 4:

SELECT (C.FIRST\_NAME || ', ' || C.SURNAME) AS "CUSTOMER", B.EMPLOYEE\_ID, D.DESCRIPTION, P.PRODUCT, B.BILL\_DATE

FROM CUSTOMER C, BILLING B, PRODUCT\_BILLING PB, DELIVERY D, PRODUCT P

WHERE C.CUSTOMER\_ID = B.CUSTOMER\_ID

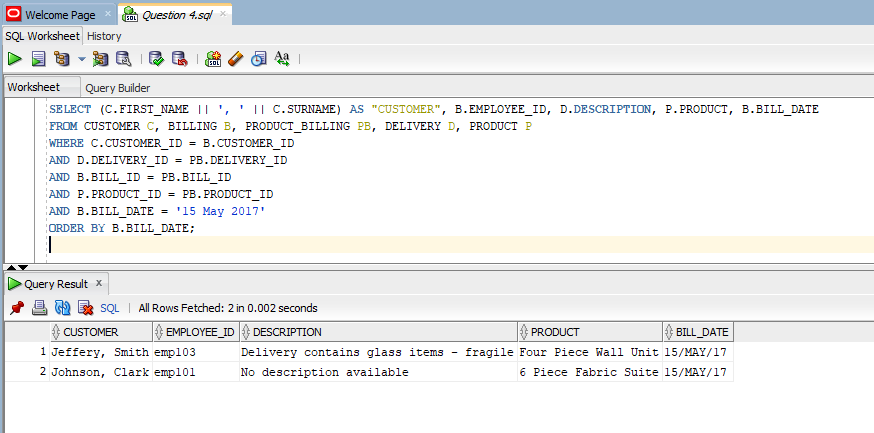
AND D.DELIVERY\_ID = PB.DELIVERY\_ID

AND B.BILL\_ID = PB.BILL\_ID

AND P.PRODUCT\_ID = PB.PRODUCT\_ID

AND B.BILL\_DATE = '15 May 2017'

ORDER BY B.EMPLOYEE\_ID DESC;



# Question 5:

CREATE OR REPLACE VIEW Inventory AS

SELECT PRODUCT, 'R' || PRICE AS "PRICE", QTY, 'R' || (PRICE \* QTY) AS "STOCK VALUE"

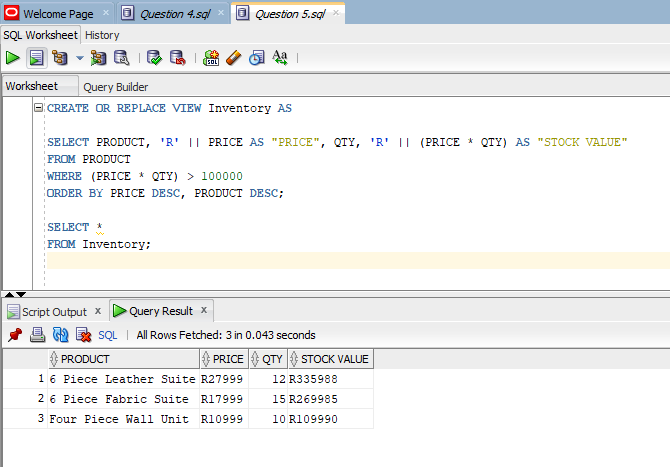
FROM PRODUCT

WHERE (PRICE \* QTY) > 100000

ORDER BY PRICE DESC, PRODUCT DESC;

SELECT \*

FROM Inventory;



# Question 6:

SET SERVEROUTPUT ON;

DECLARE

CUSTOMER VARCHAR(50);

PRODUCT VARCHAR(50);

CURSOR A IS

SELECT (C.FIRST\_NAME || ', ' || C.SURNAME), P.PRODUCT

FROM CUSTOMER C

INNER JOIN BILLING B ON C.CUSTOMER\_ID = B.CUSTOMER\_ID

INNER JOIN PRODUCT\_BILLING PB ON B.BILL\_ID = PB.BILL\_ID

INNER JOIN PRODUCT P ON PB.PRODUCT\_ID = P.PRODUCT\_ID

WHERE P.PRICE > 10000

ORDER BY P.PRODUCT DESC;

BEGIN

OPEN A;

FETCH A INTO CUSTOMER, PRODUCT;

LOOP

DBMS\_OUTPUT.PUT\_LINE('CUSTOMER: ' || chr(9) || CUSTOMER);

DBMS\_OUTPUT.PUT\_LINE('PRODUCT: ' || chr(9) || PRODUCT);

DBMS\_OUTPUT.PUT\_LINE('--------------------------------------');

FETCH A INTO CUSTOMER, PRODUCT;

EXIT WHEN A %NOTFOUND;

END LOOP;

CLOSE A;

END;



# Question 7:

SET SERVEROUTPUT ON;

DECLARE

CUSTOMER VARCHAR(50);

PRODUCT VARCHAR(50);

REASON VARCHAR(100);

CURSOR A IS

SELECT (C.FIRST\_NAME || ', ' || C.SURNAME), P.PRODUCT, R.REASON

FROM CUSTOMER C

INNER JOIN BILLING B ON C.CUSTOMER\_ID = B.CUSTOMER\_ID

INNER JOIN PRODUCT\_BILLING PB ON B.BILL\_ID = PB.BILL\_ID

INNER JOIN PRODUCT P ON PB.PRODUCT\_ID = P.PRODUCT\_ID

INNER JOIN RETURNS R ON PB.RETURN\_ID = R.RETURN\_ID

ORDER BY R.REASON;

BEGIN

OPEN A;

FETCH A INTO CUSTOMER, PRODUCT, REASON;

LOOP

DBMS\_OUTPUT.PUT\_LINE('CUSTOMER: ' || chr(9) || chr(9) || CUSTOMER);

DBMS\_OUTPUT.PUT\_LINE('PRODUCT: ' || chr(9) || chr(9) || PRODUCT);

DBMS\_OUTPUT.PUT\_LINE('RETURN REASON: ' || ' ' || REASON);

DBMS\_OUTPUT.PUT\_LINE('---------------------------------------------------');

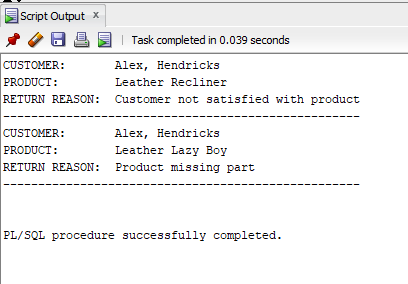
FETCH A INTO CUSTOMER, PRODUCT, REASON;

EXIT WHEN A %NOTFOUND;

END LOOP;

CLOSE A;

END;



# Question 8:

SET SERVEROUTPUT ON;

DECLARE

PRODUCT VARCHAR(50);

PRICE DECIMAL(9, 2);

PRODUCT\_TYPE VARCHAR(25);

DESCRIPTION VARCHAR(100);

DELIVERY\_DAYS INT;

CURSOR A IS

SELECT P.PRODUCT, P.PRICE, D.DESCRIPTION, (D.DELIVERY\_DATE - D.DISPATCH\_DATE)

FROM PRODUCT P

INNER JOIN PRODUCT\_BILLING PB ON P.PRODUCT\_ID = PB.PRODUCT\_ID

INNER JOIN DELIVERY D ON PB.DELIVERY\_ID = D.DELIVERY\_ID

WHERE (D.DELIVERY\_DATE - D.DISPATCH\_DATE) > 3

ORDER BY P.PRODUCT DESC, D.DESCRIPTION;

BEGIN

OPEN A;

FETCH A INTO PRODUCT, PRICE, DESCRIPTION, DELIVERY\_DAYS;

LOOP

IF PRICE >= 15000 THEN

PRODUCT\_TYPE := 'Premium Product';

ELSIF PRICE < 15000 THEN

PRODUCT\_TYPE := 'Non Premium Product';

END IF;

DBMS\_OUTPUT.PUT\_LINE('PRODUCT: ' || chr(9) || chr(9) || ' ' || PRODUCT);

DBMS\_OUTPUT.PUT\_LINE('PRICE:' || chr(9) || chr(9) || chr(9) || ' R' || PRICE);

DBMS\_OUTPUT.PUT\_LINE('DESCRIPTION: ' || chr(9) || ' ' || DESCRIPTION);

DBMS\_OUTPUT.PUT\_LINE('DELIVERY DAYS: ' || chr(9) || ' ' || DELIVERY\_DAYS);

DBMS\_OUTPUT.PUT\_LINE('PRODUCT\_CATEGORY: ' || PRODUCT\_TYPE);

DBMS\_OUTPUT.PUT\_LINE('------------------------------------------------------------------');

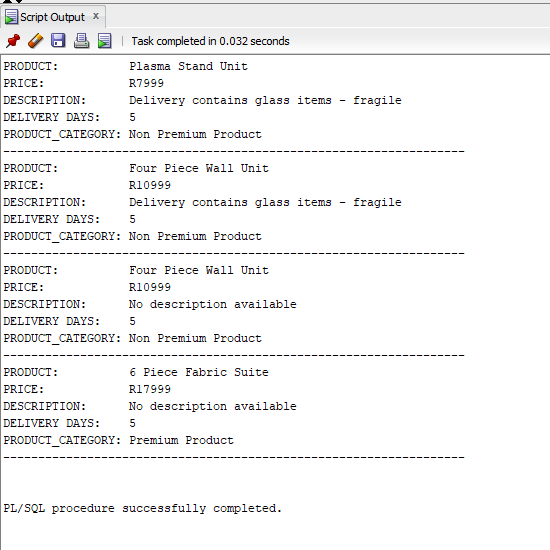
FETCH A INTO PRODUCT, PRICE, DESCRIPTION, DELIVERY\_DAYS;

EXIT WHEN A %NOTFOUND;

END LOOP;

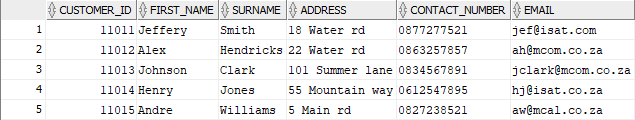
CLOSE A;

END;

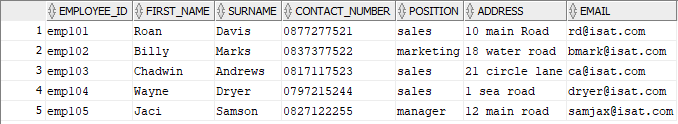


# Tables:

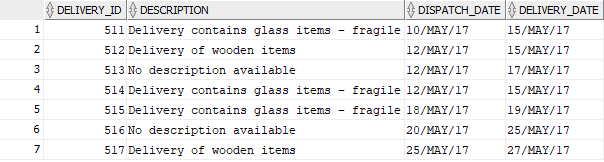
Customer:



Employee:



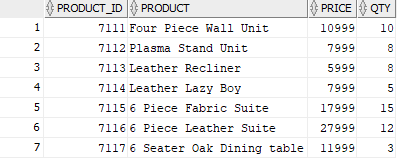
Delivery:



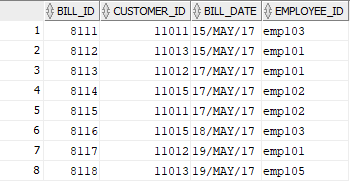
Returns:



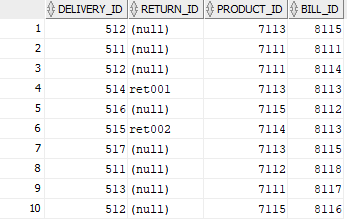
Product:



Billing:



Product\_Billing:



# Bibliography

Carlos Coronel, S. M. P. R. D. K. C., 2017. *Database Principles Fundamentals of Design, and Management.* 2 ed. Hampshire: Andrew Ashwin.

Eid, M. I., 2012. *A Learning System For Entity Relationship Modeling,* Dhahran: Department of Accounting and MIS.

Joseph V. Homan, P. J. K., 2009. *A COMPARISON OF THE RELATIONAL DATABASE MODEL AND THE ASSOCIATIVE DATABASE MODEL,* Pittsburgh: Robert Morris University.

Prabhjot, N. S., 2017. Overview of the Database Management System. *International Journal of Advanced Research in Computer Science,* 8(4), p. 8.

Radoslava Kraleva, V. K. N. S., 2018. *Design and Analysis of a Relational Database for Behavioral Experiments Data Processing,* Blagoevgrad: South-West University.

Singh, P., 2015. Studies and Analysis of Popular Database Models. *A Monthly Journal of Computer Science and Information Technology,* 4(5), p. 5.

The Independent Institute of Education, 2019. *ADVANCED DATABASES ASSIGNMENT 1.* Johannesburg: The Independent Institute of Education.