

Assignment 1

ADDB7311



September 5, 2024

ST10393280

Daniel Luke James

Table of Contents

[Question 1 2](#_Toc176472724)

[Question 2 3](#_Toc176472725)

[Question 3 7](#_Toc176472726)

[Q.3.1 7](#_Toc176472727)

[Q.3.2 7](#_Toc176472728)

[Question 4 8](#_Toc176472729)

[Q.4.1 8](#_Toc176472730)

[Q.4.2 8](#_Toc176472731)

[Question 5 10](#_Toc176472732)

[Q.5.1 10](#_Toc176472733)

[Q.5.2 10](#_Toc176472734)

[Q.5.3 10](#_Toc176472735)

[Q.5.3.1 10](#_Toc176472736)

[Q.5.3.2 11](#_Toc176472737)

[Question 6 12](#_Toc176472738)

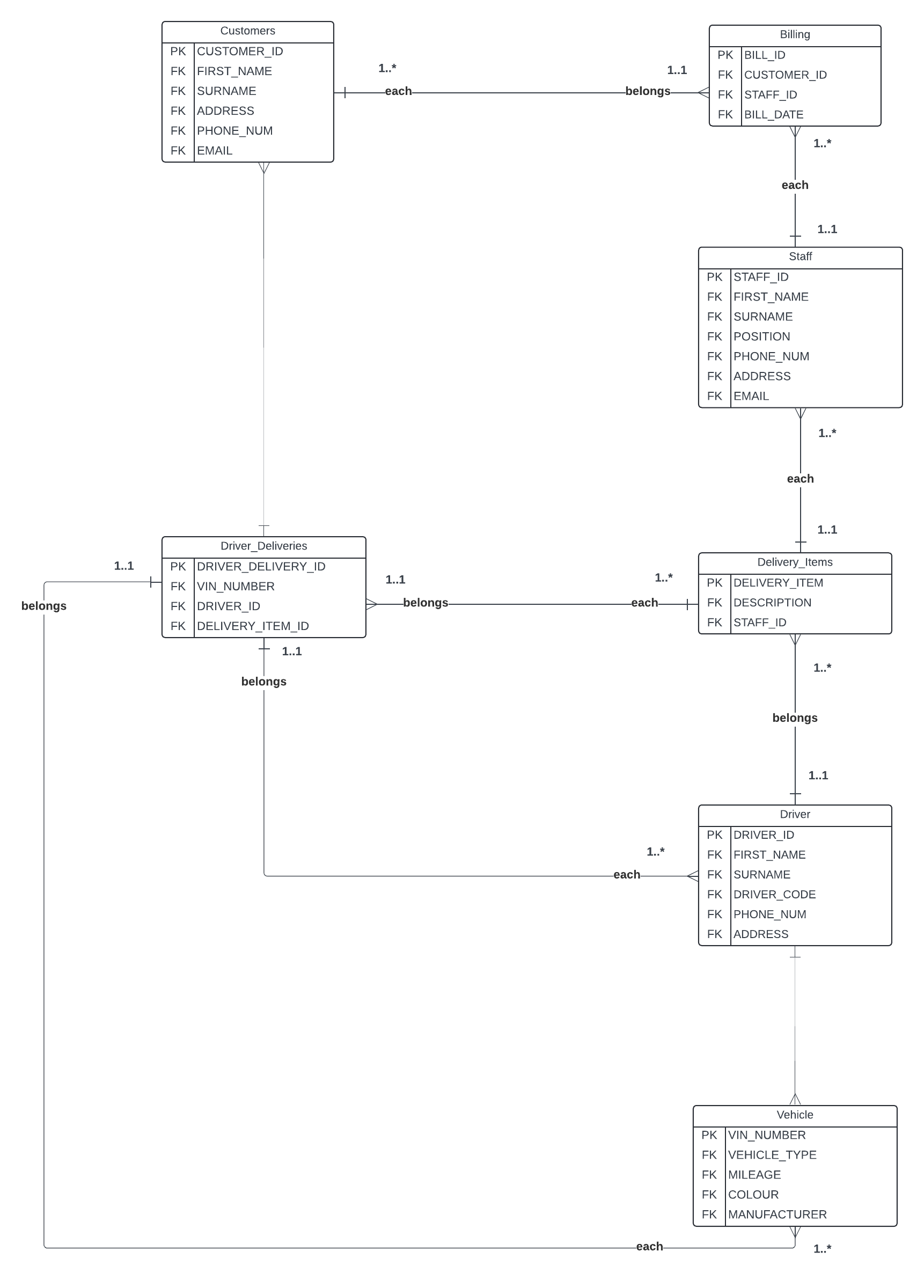
[Q.6.1 12](#_Toc176472739)

[Q.6.2 13](#_Toc176472740)

[SQL Code 15](#_Toc176472741)

[Reference List 33](#_Toc176472742)

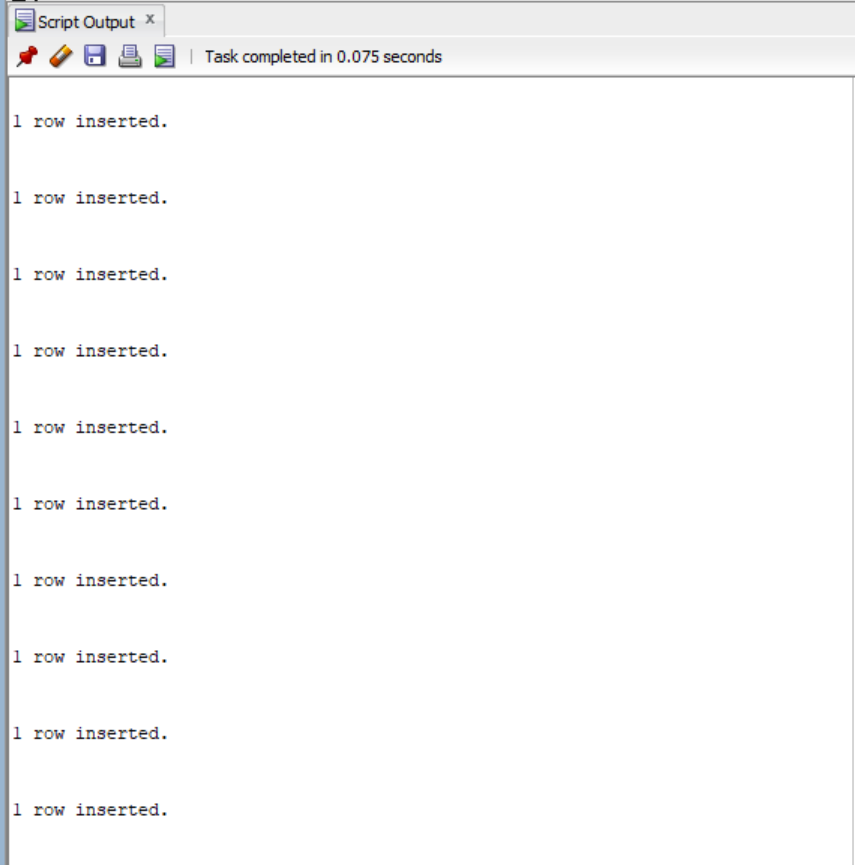
# Question 1

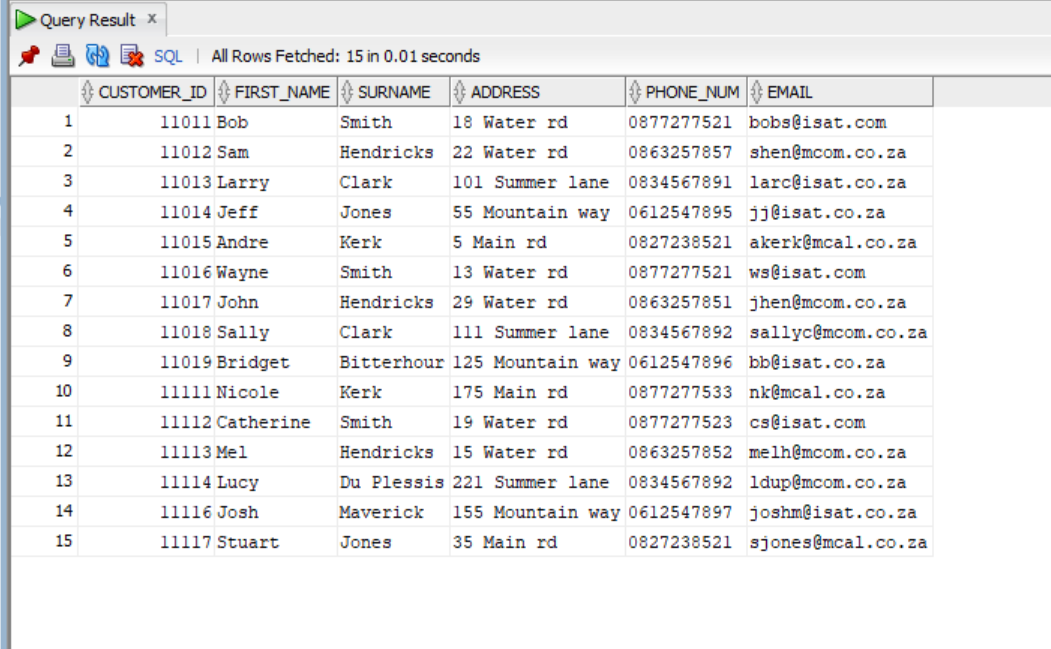
****

# Question 2

**A screenshot of a computer

Description automatically generatedTables Created:**

**Tables Inserted:**

**Customers:**

**Billing:**

**A screenshot of a computer

Description automatically generated**

**Delivery\_Items:**

**A screenshot of a computer

Description automatically generated**

**Driver:**

**A screenshot of a computer

Description automatically generated**

**Driver\_Deliveries:**

**A screenshot of a computer

Description automatically generated**

**Staff:**

A screenshot of a computer

Description automatically generated

**Vehicle:**

A screenshot of a computer

Description automatically generated

# Question 3

### Q.3.1

A screenshot of a computer

Description automatically generated

### Q.3.2

* **Conformity and Risk Control:** Restricting access to pertinent data improves trust and lowers legal liabilities while assisting in risk management and ensuring adherence to safeguarding information laws.
* **Responsibility and Execution Monitoring**: Keeping tasks distinct makes it easier to pinpoint problems or potential areas for development. It also enables accurate performance tracking.
* **Operational Efficiency:** When roles are clearly defined, work is streamlined, and productivity is increased. Staff members concentrate on assigned tasks, and drivers only have access to the delivery information they need.
* **Security of information and confidentiality:** By limiting access, you can make sure that private data is safe from breaches and unauthorized use.

# Question 4

### A screenshot of a computer Description automatically generatedQ.4.1

### Q.4.2

**Flat File Database Model:**

* Queries and data integrity can be challenging.
* Lacks connections between the records.
* Keeps information in a single file or table.
* High duplication of data and restricted scalability.

**Relational Database Model:**

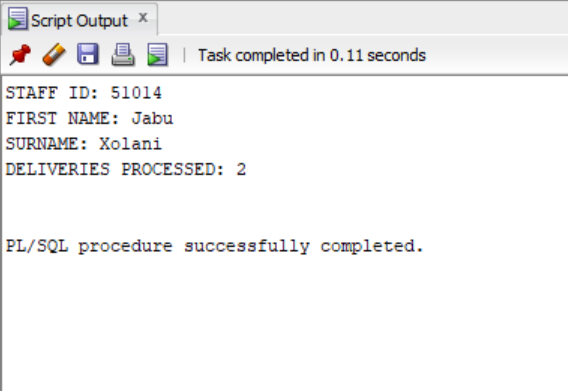
* Makes use of SQL for effective and versatile querying.
* Arranges information into several connected tables.
* Decreases redundancy and works well with big datasets.
* Upholds the relationships and constraints that preserve data integrity.

I believe a relational model will better suit CHEETAH DELIVERIES of the following:

* SQL can be used by CHEETAH DELIVERIES to execute sophisticated queries and produce comprehensive reports on operational metrics, customer interactions, and delivery performance. Operational efficiency and sound decision-making are supported by this capability.
* Relational tables allow the business to handle and evaluate its operations more effectively by connecting related data. This methodical approach improves the accessibility and organization of data.
* SQL can be used by CHEETAH DELIVERIES to execute sophisticated queries and produce comprehensive reports on operational metrics, customer interactions, and delivery performance. Operational efficiency and sound decision-making are supported by this capability.

# Question 5

### Q.5.1



### Q.5.2

* **Declaration Section:** In this section, data types defined by users that will be utilized in the PL/SQL block, such as variables, constants, and cursors, are declared.
* **Execution Section:** This section houses the primary code that carries out the operations and actions. It consists of statements that can be executed, like conditionals, loops, SQL queries, and assignments.
* **Exception Handling Section:** In this section, failures and exceptions that arise during the block's execution are handled. It offers a means of handling runtime errors and handling them gracefully.

(tutorialspoint, 2024).

### Q.5.3

### Q.5.3.1

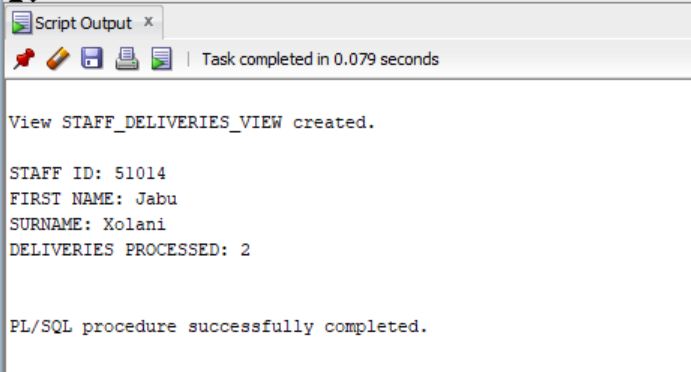
**How a View Operates:**

* A CREATE VIEW statement is used to define a view. Its foundation is a SQL query that pulls information from one or more supporting tables.
* Users view the outcome of a view's SQL query as though it were a table of data when they query a view. Every time the view is accessed, data is dynamically retrieved from the underlying tables.
* Views can be used to restrict access to specific data rows or columns, present data in a particular format, or simplify complex queries.

**Benefit:** By offering an established query that conveys the data in a particular format or subset, a view makes data access easier for management. Managers are not required to write intricate SQL queries to obtain the reports they require. This lowers the possibility of mistakes and improves process usability.

(microsoft, 2024).

### Q.5.3.2



# Question 6

### Q.6.1

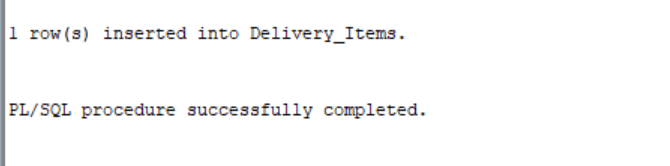
* 1. **HOW?** Oracle automatically creates implicit cursors upon the execution of a SQL statement, such as SELECT, INSERT, UPDATE, or DELETE. Simple queries that require no explicit authority over the cursor operations can benefit from the use of implicit cursors. More management of query processing is possible by utilizing the CURSOR keyword, which enables intricate calculations to be performed, repeating by means of a result set, and retrieving rows one at a time.

**WHY Use Implicit Cursor?** Implicit cursor attributes are perfect for simple operations requiring little coding effort because they enable fast and simple browsing of the current state of operations performed with DML lacking requiring explicit cursor handling.

* 1. **HOW?** Programmers can define explicit cursors explicitly by using the CURSOR keyword. This gives them greater authority over query processing, enabling them to execute complex logic, loop through an outcome set, and retrieve rows one at a time.

**WHY Use Explicit Cursor?** When processing a query result row-by-row is required, as in complicated business logic, or when retrieving multiple rows and performing iterative operations on them, explicit cursors are employed.

(tutorialspoint, 2024).



A screenshot of a computer

Description automatically generated

### Q.6.2

When it comes to Cheetah Deliveries, a sequence can be especially helpful in creating unique identifiers for records like delivery IDs, customer IDs, or transaction IDs. This way, each new entry is automatically assigned a unique ID without the need for manual input or running the risk of duplication.

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

# SQL Code

-- ST10393280 Daniel Luke James

-- Question 2

-- Create Tables

CREATE TABLE Customers (

CUSTOMER\_ID NUMBER PRIMARY KEY,

FIRST\_NAME VARCHAR2(50),

SURNAME VARCHAR2(50),

ADDRESS VARCHAR2(100),

PHONE\_NUM VARCHAR2(20),

EMAIL VARCHAR2(100)

);

CREATE TABLE Billing (

BILL\_ID NUMBER PRIMARY KEY,

CUSTOMER\_ID NUMBER REFERENCES Customers(CUSTOMER\_ID),

STAFF\_ID NUMBER,

BILL\_DATE DATE

);

CREATE TABLE Delivery\_Items (

DELIVERY\_ITEM NUMBER PRIMARY KEY,

DESCRIPTION VARCHAR2(100),

STAFF\_ID NUMBER

);

CREATE TABLE Driver (

DRIVER\_ID NUMBER PRIMARY KEY,

FIRST\_NAME VARCHAR2(50),

SURNAME VARCHAR2(50),

DRIVER\_CODE VARCHAR2(5),

PHONE\_NUM VARCHAR2(20),

ADDRESS VARCHAR2(100)

);

CREATE TABLE Staff (

STAFF\_ID NUMBER PRIMARY KEY,

FIRST\_NAME VARCHAR2(50),

SURNAME VARCHAR2(50),

POSITION VARCHAR2(50),

PHONE\_NUM VARCHAR2(20),

ADDRESS VARCHAR2(100),

EMAIL VARCHAR2(100)

);

CREATE TABLE Driver\_Deliveries (

DRIVER\_DELIVERY\_ID NUMBER PRIMARY KEY,

VIN\_NUMBER VARCHAR2(17),

DRIVER\_ID NUMBER REFERENCES Driver(DRIVER\_ID),

DELIVERY\_ITEM\_ID NUMBER REFERENCES Delivery\_Items(DELIVERY\_ITEM)

);

CREATE TABLE Vehicle (

VIN\_NUMBER VARCHAR2(17) PRIMARY KEY,

VEHICLE\_TYPE VARCHAR2(50),

MILEAGE NUMBER,

COLOUR VARCHAR2(20),

MANUFACTURER VARCHAR2(50)

);

-- Insert Data

-- Customers Table

-- Insert Data into Customers Table

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11011, 'Bob', 'Smith', '18 Water rd', '0877277521', 'bobs@isat.com');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11012, 'Sam', 'Hendricks', '22 Water rd', '0863257857', 'shen@mcom.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11013, 'Larry', 'Clark', '101 Summer lane', '0834567891', 'larc@isat.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11014, 'Jeff', 'Jones', '55 Mountain way', '0612547895', 'jj@isat.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11015, 'Andre', 'Kerk', '5 Main rd', '0827238521', 'akerk@mcal.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11016, 'Wayne', 'Smith', '13 Water rd', '0877277521', 'ws@isat.com');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11017, 'John', 'Hendricks', '29 Water rd', '0863257851', 'jhen@mcom.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11018, 'Sally', 'Clark', '111 Summer lane', '0834567892', 'sallyc@mcom.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11019, 'Bridget', 'Bitterhour', '125 Mountain way', '0612547896', 'bb@isat.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11111, 'Nicole', 'Kerk', '175 Main rd', '0877277533', 'nk@mcal.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11112, 'Catherine', 'Smith', '19 Water rd', '0877277523', 'cs@isat.com');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11113, 'Mel', 'Hendricks', '15 Water rd', '0863257852', 'melh@mcom.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11114, 'Lucy', 'Du Plessis', '221 Summer lane', '0834567892', 'ldup@mcom.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11116, 'Josh', 'Maverick', '155 Mountain way', '0612547897', 'joshm@isat.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11117, 'Stuart', 'Jones', '35 Main rd', '0827238521', 'sjones@mcal.co.za');

-- Billing Table

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(800, 11011, 51011, TO\_DATE('06-Sep-22', 'DD-Mon-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(801, 11012, 51013, TO\_DATE('07-Sep-22', 'DD-Mon-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(802, 11014, 51015, TO\_DATE('10-Nov-22', 'DD-Mon-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(803, 11015, 51012, TO\_DATE('09-Dec-22', 'DD-Mon-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(804, 11013, 51014, TO\_DATE('09-Dec-22', 'DD-Mon-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(805, 11111, 51011, TO\_DATE('06-Sep-22', 'DD-Mon-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(806, 11012, 51013, TO\_DATE('07-Sep-22', 'DD-Mon-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(807, 11014, 51015, TO\_DATE('10-Nov-22', 'DD-Mon-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(808, 11015, 51012, TO\_DATE('09-Dec-22', 'DD-Mon-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(809, 11111, 51011, TO\_DATE('06-Sep-22', 'DD-Mon-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(810, 11019, 51013, TO\_DATE('09-Dec-22', 'DD-Mon-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(811, 11011, 51011, TO\_DATE('06-Sep-22', 'DD-Mon-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(812, 11014, 51016, TO\_DATE('10-Nov-22', 'DD-Mon-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(813, 11117, 51014, TO\_DATE('09-Dec-22', 'DD-Mon-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(814, 11116, 51015, TO\_DATE('09-Dec-22', 'DD-Mon-YY'));

-- Delivery\_Items Table

INSERT INTO Delivery\_Items (DELIVERY\_ITEM, DESCRIPTION, STAFF\_ID) VALUES

(71011, 'House relocation', 51011);

INSERT INTO Delivery\_Items (DELIVERY\_ITEM, DESCRIPTION, STAFF\_ID) VALUES

(71012, 'Delivery of specialized consignments', 51017);

INSERT INTO Delivery\_Items (DELIVERY\_ITEM, DESCRIPTION, STAFF\_ID) VALUES

(71013, 'Delivery of specialized consignments', 51015);

INSERT INTO Delivery\_Items (DELIVERY\_ITEM, DESCRIPTION, STAFF\_ID) VALUES

(71014, 'Office relocation', 51012);

INSERT INTO Delivery\_Items (DELIVERY\_ITEM, DESCRIPTION, STAFF\_ID) VALUES

(71015, 'Delivery of specialized consignments', 51014);

-- Driver Table

INSERT INTO Driver (DRIVER\_ID, FIRST\_NAME, SURNAME, DRIVER\_CODE, PHONE\_NUM, ADDRESS) VALUES

(81011, 'Buthelezi', 'Marshall', 'C1', '0725698547', '18 Leopard creek');

INSERT INTO Driver (DRIVER\_ID, FIRST\_NAME, SURNAME, DRIVER\_CODE, PHONE\_NUM, ADDRESS) VALUES

(81012, 'Tina', 'Mtati', 'C', '0636984178', '12 Cape rd');

INSERT INTO Driver (DRIVER\_ID, FIRST\_NAME, SURNAME, DRIVER\_CODE, PHONE\_NUM, ADDRESS) VALUES

(81013, 'Jono', 'Mvuyisi', 'EC1', '0725648965', '15 Circle lane');

INSERT INTO Driver (DRIVER\_ID, FIRST\_NAME, SURNAME, DRIVER\_CODE, PHONE\_NUM, ADDRESS) VALUES

(81014, 'Richard', 'Smith', 'C1', '0623116598', '18 Beach rd');

INSERT INTO Driver (DRIVER\_ID, FIRST\_NAME, SURNAME, DRIVER\_CODE, PHONE\_NUM, ADDRESS) VALUES

(81015, 'Brett', 'Smith', 'EB', '0883521457', '55 Summer lane');

-- Driver\_Deliveries Table

INSERT INTO Driver\_Deliveries (DRIVER\_DELIVERY\_ID, VIN\_NUMBER, DRIVER\_ID, DELIVERY\_ITEM\_ID) VALUES

(91011, '17A55858541', 81011, 71011);

INSERT INTO Driver\_Deliveries (DRIVER\_DELIVERY\_ID, VIN\_NUMBER, DRIVER\_ID, DELIVERY\_ITEM\_ID) VALUES

(91012, '17A35858543', 81012, 71013);

INSERT INTO Driver\_Deliveries (DRIVER\_DELIVERY\_ID, VIN\_NUMBER, DRIVER\_ID, DELIVERY\_ITEM\_ID) VALUES

(91013, '17A17851545', 81011, 71015);

INSERT INTO Driver\_Deliveries (DRIVER\_DELIVERY\_ID, VIN\_NUMBER, DRIVER\_ID, DELIVERY\_ITEM\_ID) VALUES

(91014, '17A35868540', 81013, 71015);

INSERT INTO Driver\_Deliveries (DRIVER\_DELIVERY\_ID, VIN\_NUMBER, DRIVER\_ID, DELIVERY\_ITEM\_ID) VALUES

(91015, '17A15851545', 81014, 71012);

-- Staff Table

INSERT INTO Staff (STAFF\_ID, FIRST\_NAME, SURNAME, POSITION, PHONE\_NUM, ADDRESS, EMAIL) VALUES

(51011, 'Sally', 'Du Toit', 'Logistics', '0825698547', '18 Main rd', 'sdut@isat.com');

INSERT INTO Staff (STAFF\_ID, FIRST\_NAME, SURNAME, POSITION, PHONE\_NUM, ADDRESS, EMAIL) VALUES

(51012, 'Mark', 'Wright', 'CRM', '0836984178', '12 Cape way', 'mwright@isat.com');

INSERT INTO Staff (STAFF\_ID, FIRST\_NAME, SURNAME, POSITION, PHONE\_NUM, ADDRESS, EMAIL) VALUES

(51013, 'Harry', 'Sheen', 'Logistics', '0725648965', '15 Water Street', 'hsheen@isat.com');

INSERT INTO Staff (STAFF\_ID, FIRST\_NAME, SURNAME, POSITION, PHONE\_NUM, ADDRESS, EMAIL) VALUES

(51014, 'Jabu', 'Xolani', 'Logistics', '0823116598', '18 White Lane', 'jxo@isat.com');

INSERT INTO Staff (STAFF\_ID, FIRST\_NAME, SURNAME, POSITION, PHONE\_NUM, ADDRESS, EMAIL) VALUES

(51015, 'Roberto', 'Henry', 'Packaging', '0783521451', '55 Cape Street', 'rhenry@isat.com');

-- Vehicle Table

INSERT INTO Vehicle (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('17A55858541', 'Cutaway van chassis', 115352, 'RED', 'MAN');

INSERT INTO Vehicle (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('1ZA51858542', 'Flatbed truck', 315856, 'BLUE', 'ISUZU');

INSERT INTO Vehicle (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('17A35858543', 'Medium Standard Truck', 789587, 'SILVER', 'MAN');

INSERT INTO Vehicle (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('17A15851545', 'Flatbed truck', 555050, 'WHITE', 'TATA');

INSERT INTO Vehicle (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('17A35868540', 'Cutaway van chassis', 79058, 'WHITE', 'ISUZU');

-- View each tables data

SELECT \* FROM Customers;

SELECT \* FROM Billing;

SELECT \* FROM Delivery\_Items;

SELECT \* FROM Driver;

SELECT \* FROM Driver\_Deliveries;

SELECT \* FROM Staff;

SELECT \* FROM Vehicle;

-- Question 3.1

CREATE USER C##John IDENTIFIED BY Johnch2024;

GRANT CONNECT TO C##John;

GRANT SELECT ANY TABLE TO C##John;

CREATE USER C##Hannah IDENTIFIED BY Hannahch2024;

GRANT CONNECT TO C##Hannah;

GRANT INSERT ANY TABLE TO C##Hannah;

-- Question 4.1

SELECT

D.FIRST\_NAME || ', ' || D.SURNAME AS DRIVER,

D.DRIVER\_CODE AS CODE,

V.VIN\_NUMBER,

V.MILEAGE

FROM

Driver\_Deliveries DD

JOIN

Driver D ON DD.DRIVER\_ID = D.DRIVER\_ID

JOIN

Vehicle V ON DD.VIN\_NUMBER = V.VIN\_NUMBER

WHERE

V.MILEAGE < 80000;

-- Question 5.1

-- I assume there is an error with the assignment. Delivery\_Items table only has one occurrence for each Staff\_ID.

-- I inserted another line into the Delivery\_Items table, so there is two occurrences for a Staff\_ID, being '51014'.

INSERT INTO Delivery\_Items (DELIVERY\_ITEM, DESCRIPTION, STAFF\_ID) VALUES

(71016, 'Test Item', 51014);

-- PL/SQL procedure

SET SERVEROUTPUT ON;

DECLARE

v\_staff\_id Staff.STAFF\_ID%TYPE;

v\_first\_name Staff.FIRST\_NAME%TYPE;

v\_surname Staff.SURNAME%TYPE;

v\_deliveries\_processed NUMBER;

BEGIN

SELECT STAFF\_ID, FIRST\_NAME, SURNAME, DELIVERIES\_COUNT

INTO v\_staff\_id, v\_first\_name, v\_surname, v\_deliveries\_processed

FROM (

SELECT S.STAFF\_ID, S.FIRST\_NAME, S.SURNAME, COUNT(D.DELIVERY\_ITEM) AS DELIVERIES\_COUNT

FROM Staff S

JOIN Delivery\_Items D ON S.STAFF\_ID = D.STAFF\_ID

GROUP BY S.STAFF\_ID, S.FIRST\_NAME, S.SURNAME

ORDER BY COUNT(D.DELIVERY\_ITEM) DESC

)

WHERE ROWNUM = 1;

-- Display

DBMS\_OUTPUT.PUT\_LINE('STAFF ID: ' || v\_staff\_id);

DBMS\_OUTPUT.PUT\_LINE('FIRST NAME: ' || v\_first\_name);

DBMS\_OUTPUT.PUT\_LINE('SURNAME: ' || v\_surname);

DBMS\_OUTPUT.PUT\_LINE('DELIVERIES PROCESSED: ' || v\_deliveries\_processed);

END;

/

-- Question 5.3.2

-- Creating the View

CREATE OR REPLACE VIEW STAFF\_DELIVERIES\_VIEW AS

SELECT

S.STAFF\_ID,

S.FIRST\_NAME,

S.SURNAME,

COUNT(D.DELIVERY\_ITEM) AS DELIVERIES\_COUNT

FROM

Staff S

JOIN

Delivery\_Items D

ON

S.STAFF\_ID = D.STAFF\_ID

GROUP BY

S.STAFF\_ID,

S.FIRST\_NAME,

S.SURNAME;

-- Modified Code

SET SERVEROUTPUT ON;

DECLARE

v\_staff\_id Staff.STAFF\_ID%TYPE;

v\_first\_name Staff.FIRST\_NAME%TYPE;

v\_surname Staff.SURNAME%TYPE;

v\_deliveries\_processed NUMBER;

BEGIN

SELECT STAFF\_ID, FIRST\_NAME, SURNAME, DELIVERIES\_COUNT

INTO v\_staff\_id, v\_first\_name, v\_surname, v\_deliveries\_processed

FROM (

-- Ordered by Highest Count

SELECT STAFF\_ID, FIRST\_NAME, SURNAME, DELIVERIES\_COUNT

FROM STAFF\_DELIVERIES\_VIEW

ORDER BY DELIVERIES\_COUNT DESC

)

-- Select Highest Record

WHERE ROWNUM = 1;

-- Display

DBMS\_OUTPUT.PUT\_LINE('STAFF ID: ' || v\_staff\_id);

DBMS\_OUTPUT.PUT\_LINE('FIRST NAME: ' || v\_first\_name);

DBMS\_OUTPUT.PUT\_LINE('SURNAME: ' || v\_surname);

DBMS\_OUTPUT.PUT\_LINE('DELIVERIES PROCESSED: ' || v\_deliveries\_processed);

END;

/

-- Question 6.2

-- Example Of Implicit Cursor Attribute

BEGIN

-- Insert a new delivery item

INSERT INTO Delivery\_Items (DELIVERY\_ITEM, DESCRIPTION, STAFF\_ID)

VALUES (71018, 'Urgent Parcel Delivery', 51013);

-- Check if successful

IF SQL%ROWCOUNT > 0 THEN

DBMS\_OUTPUT.PUT\_LINE(SQL%ROWCOUNT || ' row(s) inserted into Delivery\_Items.');

ELSE

DBMS\_OUTPUT.PUT\_LINE('No rows were inserted.');

END IF;

END;

/

-- Example of Explicit Cursor Attribute

DECLARE

CURSOR driver\_cursor IS

SELECT D.FIRST\_NAME, D.SURNAME, V.VIN\_NUMBER, V.MILEAGE

FROM Driver\_Deliveries DD

JOIN Driver D ON DD.DRIVER\_ID = D.DRIVER\_ID

JOIN Vehicle V ON DD.VIN\_NUMBER = V.VIN\_NUMBER

WHERE V.MILEAGE < 80000;

v\_first\_name Driver.FIRST\_NAME%TYPE;

v\_surname Driver.SURNAME%TYPE;

v\_vin\_number Vehicle.VIN\_NUMBER%TYPE;

v\_mileage Vehicle.MILEAGE%TYPE;

BEGIN

OPEN driver\_cursor;

LOOP

FETCH driver\_cursor INTO v\_first\_name, v\_surname, v\_vin\_number, v\_mileage;

EXIT WHEN driver\_cursor%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE('Driver: ' || v\_first\_name || ' ' || v\_surname ||

', VIN: ' || v\_vin\_number ||

', Mileage: ' || v\_mileage);

END LOOP;

CLOSE driver\_cursor;

END;

/

-- Question 6.2

-- Create a sequences

CREATE SEQUENCE DELIVERY\_ITEM\_SEQ

START WITH 1000

INCREMENT BY 1

MINVALUE 1000

MAXVALUE 9999

NOCYCLE

CACHE 20;

CREATE SEQUENCE BILL\_ID\_SEQ

START WITH 1000

INCREMENT BY 1

MINVALUE 1000

MAXVALUE 9999

NOCYCLE

CACHE 20;

-- Example (Test Data)

INSERT INTO Delivery\_Items (DELIVERY\_ITEM, DESCRIPTION, STAFF\_ID)

VALUES (DELIVERY\_ITEM\_SEQ.NEXTVAL, 'Urgent Parcel Delivery', 51013);

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE)

VALUES (BILL\_ID\_SEQ.NEXTVAL, 11012, 51013, SYSDATE);

-- Display

SELECT DELIVERY\_ITEM\_SEQ.CURRVAL AS current\_delivery\_item\_id FROM dual;

SELECT BILL\_ID\_SEQ.CURRVAL AS current\_bill\_id FROM dual;

-- View Data

SELECT \* FROM Delivery\_Items;

SELECT \* FROM Billing;

# Reference List

* tutorialspoint. 2024. PL/SQL - Basic Syntax.

[Online] Available at:

<https://www.tutorialspoint.com/plsql/plsql_basic_syntax.htm>

[Accessed on 1 September 2024].

* microsoft.2024. Views. 19 July 2024.

[Online] Available at:

<https://learn.microsoft.com/en-us/sql/relational-databases/views/views?view=sql-server-ver16>

[Accessed on 1 September 2024].

* tutorialspoint. 2024. PL/SQL - Cursors.

[Online] Available at:

<https://www.tutorialspoint.com/plsql/plsql_cursors.htm>

[Accessed on 2 September 2024].