--question 1

-- create tables

CREATE TABLE CUSTOMER (

CUSTOMER\_ID INT PRIMARY KEY,

FIRST\_NAME VARCHAR(50),

SURNAME VARCHAR(50),

ADDRESS VARCHAR(50),

CONTACT\_NUMBER INT,

EMAIL VARCHAR(50)

);

CREATE TABLE EMPLOYEE(

EMPLOYEE\_ID VARCHAR(50) PRIMARY KEY,

FIRST\_NAME VARCHAR(50),

SURNAME VARCHAR(50),

CONTACT\_NUMBER INT,

ADDRESS VARCHAR(50),

EMAIL VARCHAR(50)

);

CREATE TABLE DONATOR(

DONATOR\_ID INT PRIMARY KEY,

FIRST\_NAME VARCHAR(50),

SURNAME VARCHAR(50),

CONTACT\_NUMBER INT,

EMAIL VARCHAR(50)

);

CREATE TABLE DONATION(

DONATION\_ID INT PRIMARY KEY,

DONATOR\_ID INT,

DONATION VARCHAR(50),

PRICE VARCHAR(50),

DONATION\_DATE DATE,

FOREIGN KEY (DONATOR\_ID) REFERENCES DONATOR(DONATOR\_ID)

);

CREATE TABLE DELIVERY(

DELIVERY\_ID INT PRIMARY KEY,

DELIVERY\_NOTES VARCHAR(50),

DISPATCH\_DATE DATE,

DELIVERY\_DATE DATE

);

CREATE TABLE RETURNS(

RETURN\_ID VARCHAR(50) PRIMARY KEY,

RETURN\_DATE DATE,

REASON VARCHAR(100),

CUSTOMER\_ID INT,

DONATION\_ID INT,

EMPLOYEE\_ID VARCHAR(50),

FOREIGN KEY (CUSTOMER\_ID) REFERENCES CUSTOMER(CUSTOMER\_ID),

FOREIGN KEY (DONATION\_ID) REFERENCES DONATION(DONATION\_ID),

FOREIGN KEY (EMPLOYEE\_ID) REFERENCES EMPLOYEE(EMPLOYEE\_ID)

);

CREATE TABLE INVOICE(

INVOICE\_NUM INT PRIMARY KEY,

CUSTOMER\_ID INT,

INVOICE\_DATE DATE,

EMPLOYEE\_ID VARCHAR(50),

DONATION\_ID INT,

DELIVERY\_ID INT,

FOREIGN KEY (EMPLOYEE\_ID) REFERENCES EMPLOYEE(EMPLOYEE\_ID),

FOREIGN KEY (DONATION\_ID) REFERENCES DONATION(DONATION\_ID),

FOREIGN KEY (DELIVERY\_ID) REFERENCES DELIVERY(DELIVERY\_ID)

);

-- populate tables

--customer table

INSERT INTO CUSTOMER VALUES (11011, 'Jack', 'Smith', '18 Water Rd', '0877277521', 'jsmith@isat.com');

INSERT INTO CUSTOMER VALUES (11012, 'Pat', 'Hendricks', '22 Water Rd', '0863257857', 'ph@mcom.co.za');

INSERT INTO CUSTOMER VALUES (11013, 'Andre', 'Clark', '101 Summer Lane', '0834567891', 'aclark@mcom.co.za');

INSERT INTO CUSTOMER VALUES (11014, 'Kevin', 'Jones', '55 Mountain way', '0612547895', 'kj@isat.co.za');

INSERT INTO CUSTOMER VALUES (11015, 'Lucy', 'Williams', '5 Main rd', '0827238521', 'lw@mcal.co.za');

--employee table

INSERT INTO EMPLOYEE VALUES ('emp101', 'Jeff', 'Davis', '0877277521', '10 main road', 'jand@isat.com');

INSERT INTO EMPLOYEE VALUES ('emp102', 'Kevin', 'Marks', '0837377522', '18 water road', 'km@isat.com');

INSERT INTO EMPLOYEE VALUES ('emp103', 'Adanya', 'Andrews', '0817117523', '21 circle lane', 'aa@isat.com');

INSERT INTO EMPLOYEE VALUES ('emp104', 'Adebayo', 'Dryer', '0797215244', '1 sea road', 'aryer@isat.com');

INSERT INTO EMPLOYEE VALUES ('emp105', 'Xolani', 'Samson', '0827122255', '12 main road', 'xosam@isat.com');

--donor table

INSERT INTO DONATOR VALUES (20111, 'Jeff', 'Watson', '0827172250', 'jwatson@ymail.com');

INSERT INTO DONATOR VALUES (20112, 'Stephen', 'Jones', '0837865670', 'joness@ymail.com');

INSERT INTO DONATOR VALUES (20113, 'James', 'Joe', '0878978650', 'jj@isat.com');

INSERT INTO DONATOR VALUES (20114, 'Kelly', 'Ross', '0826575650', 'kross@gsat.com');

INSERT INTO DONATOR VALUES (20115, 'Abraham', 'Clark', '0797656430', 'aclark@ymail.com');

--donation table

INSERT INTO DONATION VALUES (7111, 20111, 'KIC Fridge', 'R 599', TO\_DATE('01-MAY-2024', 'DD-MON-YYYY'));

INSERT INTO DONATION VALUES (7112, 20112, 'Samsung 42inch LCD', 'R 1299', TO\_DATE('03-MAY-2024', 'DD-MON-YYYY'));

INSERT INTO DONATION VALUES (7113, 20113, 'Sharp Microwave', 'R 1599', TO\_DATE('03-MAY-2024', 'DD-MON-YYYY'));

INSERT INTO DONATION VALUES (7114, 20115, '6 Seat Dining room table', 'R 799', TO\_DATE('05-MAY-2024', 'DD-MON-YYYY'));

INSERT INTO DONATION VALUES (7115, 20114, 'Lazyboy Sofa', 'R 1199', TO\_DATE('07-MAY-2024', 'DD-MON-YYYY'));

INSERT INTO DONATION VALUES (7116, 20113, 'JVC Surround Sound System', 'R 179', TO\_DATE('09-MAY-2024', 'DD-MON-YYYY'));

--delivery table

INSERT INTO DELIVERY VALUES (511, 'Double packaging requested', TO\_DATE('10-MAY-2024', 'DD-MON-YYYY'), TO\_DATE('15-MAY-2024', 'DD-MON-YYYY'));

INSERT INTO DELIVERY VALUES (512, 'Delivery to work address', TO\_DATE('12-MAY-2024', 'DD-MON-YYYY'), TO\_DATE('15-MAY-2024', 'DD-MON-YYYY'));

INSERT INTO DELIVERY VALUES (513, 'Signature required', TO\_DATE('12-MAY-2024', 'DD-MON-YYYY'), TO\_DATE('17-MAY-2024', 'DD-MON-YYYY'));

INSERT INTO DELIVERY VALUES (514, 'No notes', TO\_DATE('12-MAY-2024', 'DD-MON-YYYY'), TO\_DATE('15-MAY-2024', 'DD-MON-YYYY'));

INSERT INTO DELIVERY VALUES (515, 'Birthday present wrapping required', TO\_DATE('18-MAY-2024', 'DD-MON-YYYY'), TO\_DATE('19-MAY-2024', 'DD-MON-YYYY'));

INSERT INTO DELIVERY VALUES (516, 'Delivery to work address', TO\_DATE('20-MAY-2024', 'DD-MON-YYYY'), TO\_DATE('25-MAY-2024', 'DD-MON-YYYY'));

--returns table

INSERT INTO RETURNS VALUES ('ret001', TO\_DATE('25-MAY-2024', 'DD-MON-YYYY'), 'Customer not satisfied with product', 11011, 7116, 'emp101');

INSERT INTO RETURNS VALUES ('ret002', TO\_DATE('25-MAY-2024', 'DD-MON-YYYY'), 'Product had broken section', 11013, 7114, 'emp103');

--invoice table

INSERT INTO INVOICE VALUES (8111, 11011, TO\_DATE('15-MAY-2024', 'DD-MON-YYYY'), 'emp103', 7111, 511);

INSERT INTO INVOICE VALUES (8112, 11013, TO\_DATE('15-MAY-2024', 'DD-MON-YYYY'), 'emp101', 7114, 512);

INSERT INTO INVOICE VALUES (8113, 11012, TO\_DATE('17-MAY-2024', 'DD-MON-YYYY'), 'emp101', 7112, 513);

INSERT INTO INVOICE VALUES (8114, 11015, TO\_DATE('17-MAY-2024', 'DD-MON-YYYY'), 'emp102', 7113, 514);

INSERT INTO INVOICE VALUES (8115, 11011, TO\_DATE('17-MAY-2024', 'DD-MON-YYYY'), 'emp102', 7115, 515);

INSERT INTO INVOICE VALUES (8116, 11015, TO\_DATE('18-MAY-2024', 'DD-MON-YYYY'), 'emp103', 7116, 516);

--view all data stored in the tables

SELECT \* FROM CUSTOMER;

SELECT \* FROM EMPLOYEE;

SELECT \* FROM DONATOR;

SELECT \* FROM DONATION;

SELECT \* FROM DELIVERY;

SELECT \* FROM RETURNS;

SELECT \* FROM INVOICE;

--question 2

-- sql query to generate a report containing customer name, employee id, delivery notes, donation purchased and the invoice number

SELECT

c.FIRST\_NAME || ' ' || c.SURNAME AS CUSTOMER\_NAME,

i.EMPLOYEE\_ID,

d.DELIVERY\_NOTES,

don.DONATION AS DONATION\_PURCHASED,

i.INVOICE\_NUM,

i.INVOICE\_DATE

FROM

INVOICE i

JOIN

CUSTOMER c ON i.CUSTOMER\_ID = c.CUSTOMER\_ID

JOIN

DELIVERY d ON i.DELIVERY\_ID = d.DELIVERY\_ID

JOIN

DONATION don ON i.DONATION\_ID = don.DONATION\_ID

WHERE

i.INVOICE\_DATE > TO\_DATE('16-MAY-2024', 'DD-MON-YYYY');

-- question 3

--create new FUNDING table

CREATE TABLE FUNDING(

FUNDING\_ID NUMBER GENERATED ALWAYS AS IDENTITY,

FUNDER VARCHAR(50),

FUNDING\_AMOUNT INT

);

--populate table

INSERT INTO FUNDING (FUNDER, FUNDING\_AMOUNT) VALUES('Yadav Priaram', 10000);

INSERT INTO FUNDING (FUNDER, FUNDING\_AMOUNT) VALUES('Adam Smith', 50000);

--display data in the table

SELECT \* FROM FUNDING;

--this solution will create the FUNCTION table and insert the FUNDING\_ID, FUNDER, and FUNDING\_AMOUNT values.

--the solution will auto generate the FUNDING\_ID number making all ID's unique.

--the solution inserts test data into the table to show how the values are stored and how the ID is generated.

--question 4

--PL/SQL query to display customer name, donation purchased, donation price and the reason why the donation was returned.

DECLARE

v\_CUSTOMER\_NAME VARCHAR2(100);

v\_DONATION\_PURCHASED VARCHAR2(100);

v\_DONATION\_PRICE NUMBER;

v\_RETURN\_REASON VARCHAR2(200);

CURSOR cur\_returns IS

SELECT

c.FIRST\_NAME || ' ' || c.SURNAME AS CUSTOMER\_NAME,

don.DONATION AS DONATION\_PURCHASED,

TO\_NUMBER(REPLACE(don.PRICE, 'R', '')) AS DONATION\_PRICE,

r.REASON AS RETURN\_REASON

FROM

RETURNS r

JOIN

CUSTOMER c ON r.CUSTOMER\_ID = c.CUSTOMER\_ID

JOIN

DONATION don ON r.DONATION\_ID = don.DONATION\_ID;

BEGIN

FOR rec IN cur\_returns LOOP

v\_CUSTOMER\_NAME := rec.CUSTOMER\_NAME;

v\_DONATION\_PURCHASED := rec.DONATION\_PURCHASED;

v\_DONATION\_PRICE := rec.DONATION\_PRICE;

v\_RETURN\_REASON := rec.RETURN\_REASON;

--display the results in the DBMS Output window

dbms\_output.put\_line('CUSTOMER: ' || v\_CUSTOMER\_NAME);

dbms\_output.put\_line('DONATION PURCHASED: ' || v\_DONATION\_PURCHASED);

dbms\_output.put\_line('PRICE: R ' || v\_DONATION\_PRICE);

dbms\_output.put\_line('RETURN REASON: ' || v\_RETURN\_REASON);

dbms\_output.put\_line('--------------------------------------------------');

END LOOP;

dbms\_output.put\_line('PL/SQL procedure successfully completed.');

END;

--question 5

--PL/SQL query to display customer’s name, employee name, donation, dispatch date and the delivery date

DECLARE

V\_CUSTOMER\_NAME VARCHAR2(100);

V\_EMPLOYEE\_NAME VARCHAR2(100);

V\_DONATION VARCHAR2(100);

V\_DISPATCH\_DATE DATE;

V\_DELIVERY\_DATE DATE;

V\_DAYS\_TO\_DELIVERY NUMBER;

CURSOR cur\_delivery IS

SELECT

c.FIRST\_NAME || ' ' || c.SURNAME AS CUSTOMER\_NAME,

e.FIRST\_NAME || ' ' || e.SURNAME AS EMPLOYEE\_NAME,

don.DONATION AS DONATION,

TO\_DATE(del.DISPATCH\_DATE, 'DD/MON/YYYY') AS DISPATCH\_DATE,

TO\_DATE(del.DELIVERY\_DATE, 'DD/MON/YYYY') AS DELIVERY\_DATE,

TO\_DATE(del.DELIVERY\_DATE, 'DD/MON/YYYY') - TO\_DATE(del.DISPATCH\_DATE, 'DD/MON/YYYY') AS DAYS\_TO\_DELIVERY

FROM

CUSTOMER c

JOIN

INVOICE inv ON c.CUSTOMER\_ID = inv.CUSTOMER\_ID

JOIN

EMPLOYEE e ON inv.EMPLOYEE\_ID = e.EMPLOYEE\_ID

JOIN

DONATION don ON inv.DONATION\_ID = don.DONATION\_ID

JOIN

DELIVERY del ON inv.DELIVERY\_ID = del.DELIVERY\_ID

WHERE

c.CUSTOMER\_ID = 11011;

BEGIN

FOR rec IN cur\_delivery LOOP

v\_CUSTOMER\_NAME := rec.CUSTOMER\_NAME;

v\_EMPLOYEE\_NAME := rec.EMPLOYEE\_NAME;

v\_DONATION := rec.DONATION;

v\_DISPATCH\_DATE := rec.DISPATCH\_DATE;

v\_DELIVERY\_DATE := rec.DELIVERY\_DATE;

v\_DAYS\_TO\_DELIVERY := rec.DAYS\_TO\_DELIVERY;

--display the results in the DBMS Output window

dbms\_output.put\_line('CUSTOMER: ' || v\_CUSTOMER\_NAME);

dbms\_output.put\_line('EMPLOYEE: ' || v\_EMPLOYEE\_NAME);

dbms\_output.put\_line('DONATION: ' || v\_DONATION);

dbms\_output.put\_line('DISPATCH DATE: ' || TO\_CHAR(v\_DISPATCH\_DATE, 'DD/MON/YY'));

dbms\_output.put\_line('DELIVERY DATE: ' || TO\_CHAR(v\_DELIVERY\_DATE, 'DD/MON/YY'));

dbms\_output.put\_line('DAYS TO DELIVERY: ' || v\_DAYS\_TO\_DELIVERY);

dbms\_output.put\_line('--------------------------------------------------');

END LOOP;

END;

--question 6

--PL/SQL query to display customer name and total amount spent by each customer on the purchases

DECLARE

CURSOR customer\_cursor IS

SELECT

c.FIRST\_NAME,

c.SURNAME,

SUM(TO\_NUMBER(REGEXP\_SUBSTR(don.PRICE, '[0-9]+'))) AS AMOUNT,

CASE

WHEN SUM(TO\_NUMBER(REGEXP\_SUBSTR(don.PRICE, '[0-9]+'))) >= 1500 THEN '\*\*\*'

ELSE NULL

END AS RATING

FROM

CUSTOMER c

JOIN

INVOICE inv ON c.CUSTOMER\_ID = inv.CUSTOMER\_ID

JOIN

DONATION don ON inv.DONATION\_ID = don.DONATION\_ID

GROUP BY

c.FIRST\_NAME, c.SURNAME;

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

v\_surname CUSTOMER.SURNAME%TYPE;

v\_amount NUMBER;

v\_rating VARCHAR2(3);

BEGIN

FOR customer\_record IN customer\_cursor LOOP

v\_first\_name := customer\_record.FIRST\_NAME;

v\_surname := customer\_record.SURNAME;

v\_amount := customer\_record.AMOUNT;

v\_rating := customer\_record.RATING;

--display results in the DBMS Output window

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

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

DBMS\_OUTPUT.PUT\_LINE('AMOUNT: R ' || v\_amount || v\_rating);

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

END LOOP;

END;

--question 7

--Q 7.1

--example of the %TYPE attribute

DECLARE

v\_funder\_name FUNDING.FUNDER%TYPE;

v\_funding\_amount FUNDING.FUNDING\_AMOUNT%TYPE;

BEGIN

v\_funder\_name := 'Yadav Priaram';

v\_funding\_amount := 5000;

dbms\_output.put\_line('Funder Name: '|| v\_funder\_name);

dbms\_output.put\_line('Funding Amount: R ' || v\_funding\_amount);

END;

/

--Q7.2

--Example of the %ROWTYPE attribute

DECLARE

v\_funding\_record FUNDING%ROWTYPE;

BEGIN

SELECT \*

INTO v\_funding\_record

FROM FUNDING

WHERE FUNDING\_ID = 1;

dbms\_output.put\_line('Funder Name: ' || v\_funding\_record.FUNDER);

dbms\_output.put\_line('Funding Amount: R ' || v\_funding\_record.FUNDING\_AMOUNT);

END;

/

--Q7.3

--for this question, i will create a user defined excpetion which will ensure that donations are a minimum of R10

DECLARE

V\_DONATION\_AMOUNT NUMBER := 30; --INSERT DONATION AMOUNT

INSUFFICIENT\_DONATION EXCEPTION;

BEGIN

IF V\_DONATION\_AMOUNT < 10 THEN

RAISE INSUFFICIENT\_DONATION;

END IF;

DBMS\_OUTPUT.PUT\_LINE('Thank you for your donation: R ' || v\_donation\_amount);

EXCEPTION

WHEN INSUFFICIENT\_DONATION THEN

DBMS\_OUTPUT.PUT\_LINE('Error: Kindly note that the minimum donation amount is R10.');

END;

/

--Question 8

--sql query to display the customer name and total amount spent by each customer on the purchases with a rating

SELECT

c.FIRST\_NAME,

c.SURNAME,

SUM(TO\_NUMBER(REPLACE(don.PRICE, 'R', ''))) AS TOTAL\_AMOUNT\_SPENT,

CASE

WHEN SUM(TO\_NUMBER(REPLACE(don.PRICE, 'R ', ''))) >= 1500 THEN '\*\*\*'

WHEN SUM(TO\_NUMBER(REPLACE(don.PRICE, 'R ', ''))) BETWEEN 1000 AND 1400 THEN '\*\*'

ELSE '\*'

END AS CUSTOMER\_RATING

FROM

CUSTOMER c

JOIN

INVOICE inv ON c.CUSTOMER\_ID = inv.CUSTOMER\_ID

JOIN

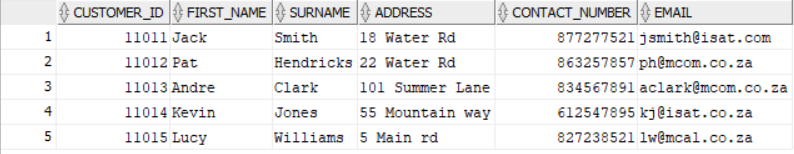
DONATION don ON inv.DONATION\_ID = don.DONATION\_ID

GROUP BY

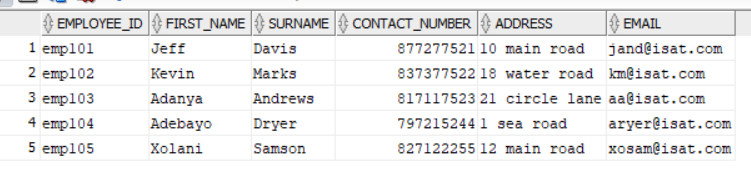
c.FIRST\_NAME, c.SURNAME

Screenshots

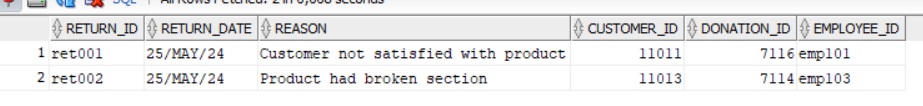
Question 1

A screenshot of a computer

Description automatically generatedA screenshot of a computer

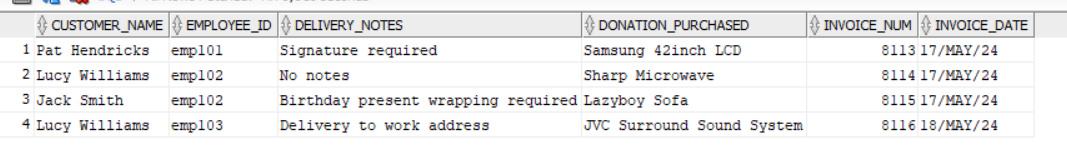
Description automatically generated

A screenshot of a computer

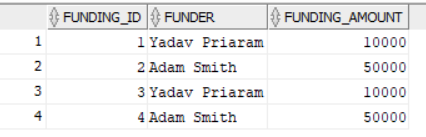
Description automatically generatedA screenshot of a computer

Description automatically generated

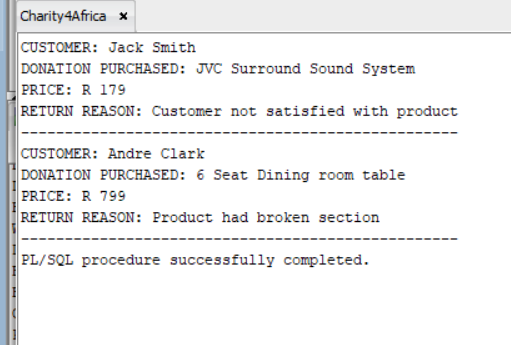
Question 2



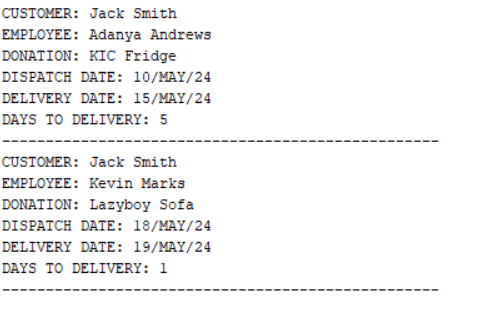
Question 3



Question 4



Question 5



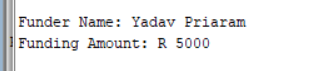
Question 6

A close-up of a number

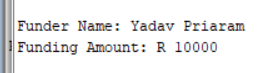
Description automatically generated

Question 7

Q7.1



Q7.2



Q7.3

Both scenarios

A close up of text

Description automatically generated

Question 8

A screenshot of a computer

Description automatically generated

Reference list

(No date) *%ROWTYPE attribute*. Available at: <https://docs.oracle.com/cd/B12037_01/appdev.101/b10807/13_elems042.htm> (Accessed: 06 October 2024).

*Loading...: Sap help portal* (no date) *loading... | SAP Help Portal*. Available at: <https://help.sap.com/docs/SAP_SQL_Anywhere/93079d4ba8e44920ae63ffb4def91f5b/817fb5486ce21014a388d42b2805512a.html#:~:text=The%20%25TYPE%20attribute%20sets%20the,type%20information%20from%20system%20tables>. (Accessed: 06 October 2024).

*Oracle Identity Column: A step-by-step guide with examples* (2023) *Oracle Tutorial*. Available at: <https://www.oracletutorial.com/oracle-basics/oracle-identity-column/> (Accessed: 06 October 2024).

*PL/SQL - exceptions* (no date) *Tutorialspoint*. Available at: <https://www.tutorialspoint.com/plsql/plsql_exceptions.htm> (Accessed: 06 October 2024).

*PL/SQL - procedures* (no date) *Tutorialspoint*. Available at: <https://www.tutorialspoint.com/plsql/plsql_procedures.htm> (Accessed: 06 October 2024).

*SQL Reference Guide* (2022) *Oracle Help Center*. Available at: <https://docs.oracle.com/en/database/other-databases/nosql-database/21.2/sqlreferencefornosql/identity-column.html#:~:text=CREATE%20TABLE%20T1%20(%20id%20INTEGER,primary%20key%20for%20table%20T>. (Accessed: 06 October 2024).

*W3schools.com* (no date) *SQL Stored Procedures*. Available at: <https://www.w3schools.com/sql/sql_stored_procedures.asp> (Accessed: 06 October 2024).