Information Systems 2C

Student Number:	ST10440981
Programme Code:	BCAD2
Module Lecturer:	Mick
Module Code:	INSY7213
Date of Submission:	26-09-2025

I hereby declare that I did not plagiarise the content of this assignment and that this is my own work.

Table of Contents

Question 1.....pg3

Question 2.....pg4

Question 3.1.....pg11

Question 3.2.....pg13

Question 4.1.....pg14

Question 4.2.....pg15

Question 5.1.....pg16

Question 5.2.....pg18

Question 5.3.1.....pg18

Question 5.3.2.....pg19

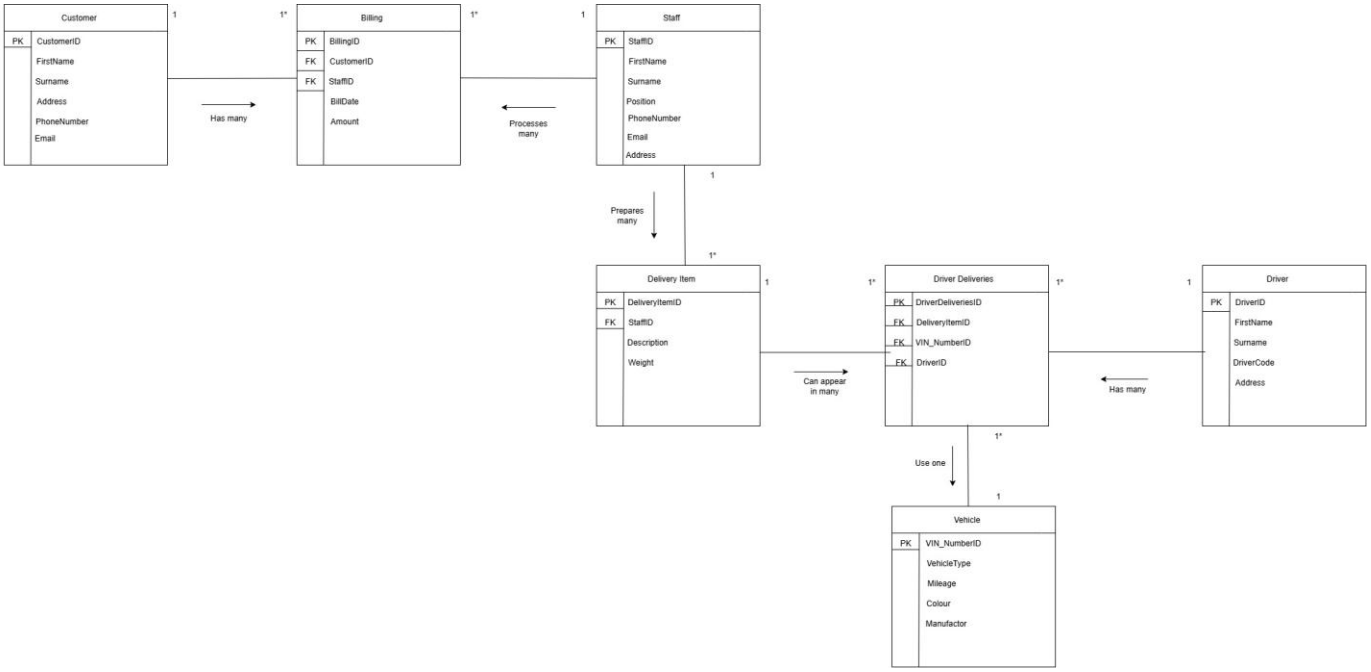
Question 6.1.....pg20

Question 6.2.....pg24

Reference List.....pg26

GitHub Link: https://github.com/anothile101/INSY2C-Practical-Assignment-1.git

Question 1



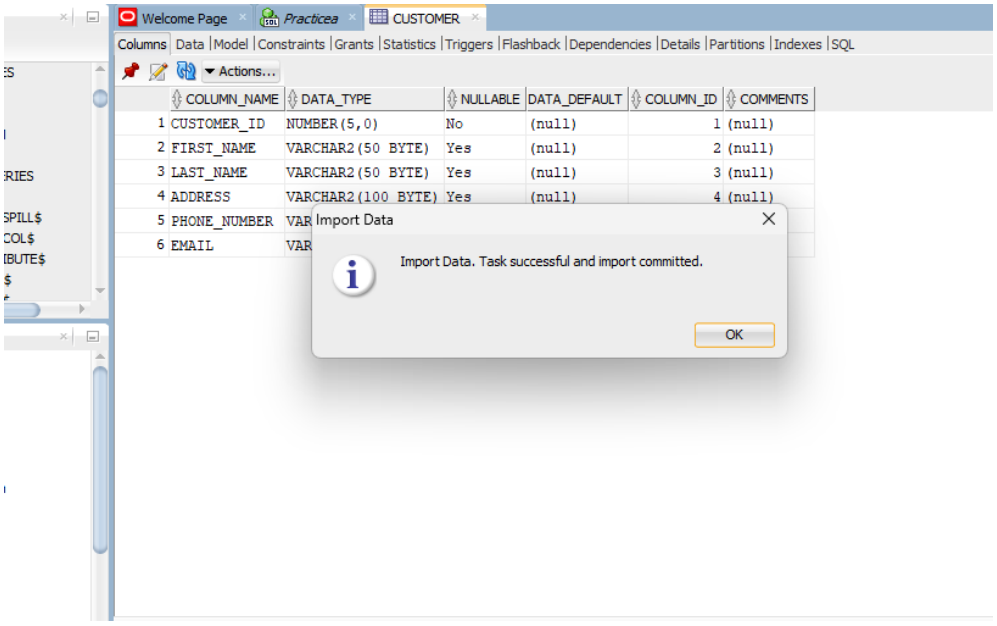
Question 2

```
-- CUSTOMER TABLE
CREATE TABLE Customer (
   Customer_ID   NUMBER(5) PRIMARY KEY,
   First_Name    VARCHAR2(50),
   Surname       VARCHAR2(50),
   Address       VARCHAR2(100),
   Phone_Num     VARCHAR2(20),
   Email         VARCHAR2(100)
);

-- STAFF TABLE
CREATE TABLE Staff (
   Staff_ID     NUMBER(5) PRIMARY KEY,
   First_Name   VARCHAR2(50),
   Surname      VARCHAR2(50),
   Position     VARCHAR2(50),
   Phone_Num    VARCHAR2(20),
   Address      VARCHAR2(100),
   Email        VARCHAR2(100)
);

-- BILLING TABLE
CREATE TABLE Billing (
   Bill_ID      NUMBER(5) PRIMARY KEY,
   Customer_ID  NUMBER(5),
   Staff_ID     NUMBER(5),
   Bill_Date    DATE,
   FOREIGN KEY (Customer_ID) REFERENCES Customer(Customer_ID),
```

```
  FOREIGN KEY (Staff_ID) REFERENCES Staff(Staff_ID)
);

-- DELIVERY_ITEM TABLE
CREATE TABLE Delivery_Item (
  Delivery_Item_ID NUMBER(5) PRIMARY KEY,
  Description     VARCHAR2(100),
  Staff_ID        NUMBER(5),
  FOREIGN KEY (Staff_ID) REFERENCES Staff(Staff_ID)
);

-- DRIVER TABLE
CREATE TABLE Driver (
  Driver_ID    NUMBER(5) PRIMARY KEY,
  First_Name   VARCHAR2(50),
  Surname      VARCHAR2(50),
  Driver_Code  VARCHAR2(10),
  Phone_Num    VARCHAR2(20),
  Address      VARCHAR2(100)
);

-- VEHICLE TABLE
CREATE TABLE Vehicle (
  VIN_Number   VARCHAR2(20) PRIMARY KEY,
  Vehicle_Type VARCHAR2(50),
  Mileage      NUMBER,
  Colour       VARCHAR2(20),
  Manufacturer VARCHAR2(50)
);
```

```
-- DRIVER_DELIVERIES TABLE
CREATE TABLE Driver_Deliveries (
   Driver_Delivery_ID NUMBER(5) PRIMARY KEY,
   VIN_Number       VARCHAR2(20),
   Driver_ID        NUMBER(5),
   Delivery_Item_ID   NUMBER(5),
   FOREIGN KEY (VIN_Number) REFERENCES Vehicle(VIN_Number),
   FOREIGN KEY (Driver_ID) REFERENCES Driver(Driver_ID),
   FOREIGN KEY (Delivery_Item_ID) REFERENCES Delivery_Item(Delivery_Item_ID)
);
```

Output of CSV imported data



Customer Table Output

Query Result — All Rows Fetched: 15 in 0,012 seconds

	CUSTOMER_ID	FIRST_NAME	SURNAME	ADDRESS	PHONE_NUM	EMAIL
1	11011	Bob	Smith	18 Water rd	0877277521	bobs@isat.com
2	11012	Sam	Hendricks	22 Water rd	0863257857	shen@mcom.co.za
3	11013	Larry	Clark	101 Summer lane	0834567891	larc@mcom.co.za
4	11014	Jeff	Jones	55 Mountain way	0612547895	jj@isat.co.za
5	11015	Andre	Kerk	5 Main rd	0827238521	akerk@mcal.co.za
6	11016	Wayne	Smith	13 Water rd	0877277522	ws@isat.com
7	11017	John	Hendricks	29 Water rd	0863257851	jhen@mcom.co.za
8	11018	Sally	Clark	111 Summer lane	0834567892	sallyc@mcom.co.za
9	11019	Bridget	Bitterhour	125 Mountain way	0612547896	bb@isat.co.za
10	11111	Nicole	Kerk	175 Main rd	0827238529	nk@mcal.co.za
11	11112	Catherine	Smith	19 Water rd	0877277523	cath@isat.com
12	11113	Mel	Hendricks	5 Water rd	0863257852	melh@mcom.co.za
13	11114	Lucy	Du Plessis	221 Summer lane	0834567892	ldup@mcom.co.za
14	11116	Josh	Maverick	155 Mountain way	0612547897	joshm@isat.co.za
15	11117	Stuart	Jones	35 Main rd	0827238521	sjones@mcal.co.za

Staff Table Output

Query Result — All Rows Fetched: 10 in 0,058 seconds

	STAFF_ID	FIRST_NAME	SURNAME	POSITION	PHONE_NUM	ADDRESS	EMAIL
1	51011	Sally	Du Toit	Logistics	0825698547	18 Main rd	sdut@isat.com
2	51012	Mark	Wright	CRM	0836984178	12 Cape Way	mwright@isat.com
3	51013	Harry	Sheen	Logistics	0725648965	15 Water Street	hsheen@isat.com
4	51014	Jabu	Xolani	Logistics	0823116598	18 White Lane	jxo@isat.com
5	51015	Roberto	Henry	Packaging	0783521451	55 Cape Street	rhenry@isat.com
6	51016	Pat	Durant	Logistics	0825698542	1 Main rd	pd@isat.com
7	51017	Steve	Maritz	CRM	0836984173	2 Cape Way	sm@isat.com
8	51018	Maxwell	Dube	Logistics	0725648964	5 Water Street	max@isat.com
9	51019	Shane	Mane	Logistics	0823116595	8 White Lane	smane@isat.com
10	51111	Bob	Truth	Packaging	0783521456	35 Cape Street	btruth@isat.com

Billing Table Output

Query Result

All Rows Fetched: 24 in 0,025 seconds

	BILL_ID	CUSTOMER_ID	STAFF_ID	BILL_DATE
1	800	11011	51011	06/SEP/22
2	801	11012	51013	07/SEP/22
3	802	11014	51015	10/NOV/22
4	803	11015	51012	09/DEC/22
5	804	11013	51014	09/DEC/22
6	805	11111	51011	06/SEP/22
7	806	11012	51013	07/SEP/22
8	807	11014	51015	10/NOV/22
9	808	11015	51012	09/DEC/22
10	809	11113	51018	09/DEC/22
11	810	11011	51011	06/SEP/22
12	811	11012	51013	07/SEP/22
13	812	11014	51016	10/NOV/22
14	813	11117	51012	09/DEC/22
15	814	11013	51014	09/DEC/22
16	815	11012	51111	06/SEP/22
17	816	11012	51019	07/SEP/22

Running : Closing Results

Billing Table Output Continued.

Query Result

All Rows Fetched: 24 in 0,025 seconds

	BILL_ID	CUSTOMER_ID	STAFF_ID	BILL_DATE
8	807	11014	51015	10/NOV/22
9	808	11015	51012	09/DEC/22
10	809	11113	51018	09/DEC/22
11	810	11011	51011	06/SEP/22
12	811	11012	51013	07/SEP/22
13	812	11014	51016	10/NOV/22
14	813	11117	51012	09/DEC/22
15	814	11013	51014	09/DEC/22
16	815	11012	51111	06/SEP/22
17	816	11012	51019	07/SEP/22
18	817	11014	51015	10/NOV/22
19	818	11112	51012	09/DEC/22
20	819	11013	51014	09/DEC/22
21	820	11116	51019	09/DEC/22
22	821	11011	51011	25/SEP/25
23	822	11011	51011	25/SEP/25
24	823	11011	51011	25/SEP/25

Running : Closing Results

Line 8 Column 23

Delivery Item Table Output

Query Result

All Rows Fetched: 5 in 0,005 seconds

	DELIVERY_ITEM_ID	DESCRIPTION	STAFF_ID
1	71011	House relocation	51011
2	71012	Delivery of specialized consignments	51017
3	71013	Delivery of specialized consignments	51015
4	71014	Office relocation	51012
5	71015	Delivery of specialized consignments	51014

Running : Closing Results | Line 11 Column 29

Vehicle Table Output

Query Result

All Rows Fetched: 20 in 0,009 seconds

	VIN_NUMBER	VEHICLE_TYPE	MILEAGE	COLOUR	MANUFACTURER
1	1ZA55858541	Cutaway van chassis	115352	RED	MAN
2	1ZA51858542	Flatbed truck	315856	BLUE	ISUZU
3	1ZA35858543	Medium Standard Truck	789587	SILVER	MAN
4	1ZA15851545	Flatbed truck	555050	WHITE	TATA
5	1ZA35868540	Cutaway van chassis	80058	WHITE	ISUZU
6	1ZA65858541	Cutaway van chassis	215352	RED	MAN
7	1ZA61858542	Flatbed truck	215856	BLUE	ISUZU
8	1ZA65858543	Medium Standard Truck	889587	SILVER	MERC
9	1ZA65851545	Flatbed truck	155050	WHITE	MAN
10	1ZA65868540	Cutaway van chassis	19058	WHITE	ISUZU
11	1ZA75858541	Cutaway van chassis	315352	RED	MAN
12	1ZA71858542	Flatbed truck	115856	BLUE	ISUZU
13	1ZA75858543	Medium Standard Truck	989587	SILVER	MAN
14	1ZA17851545	Flatbed truck	755050	WHITE	TATA
15	1ZA75868540	Cutaway van chassis	29058	WHITE	ISUZU
16	1ZA85858541	Cutaway van chassis	515352	RED	MERC
17	1ZA81858542	Flatbed truck	715856	BLUE	ISUZU

Running : Closing Results

Driver Table Output

Query Result

All Rows Fetched: 5 in 0,006 seconds

	DRIVER_ID	FIRST_NAME	SURNAME	DRIVER_CODE	PHONE_NUM	ADDRESS
1	81011	Buthelezi	Marshall	C1	0725698547	18 Leopard creek
2	81012	Tina	Mtati	C	0636984178	12 Cape rd
3	81013	Jono	Mvuyisi	EC1	0725648965	15 Circle lane
4	81014	Richard	Smith	C1	0623116598	18 Beach rd
5	81015	Brett	Smith	EB	0883521457	55 Summer lane

Running : Closing Results

Driver Deliveries Table Output

Query Result

All Rows Fetched: 5 in 0,008 seconds

	DRIVER_DELIVERY_ID	VIN_NUMBER	DRIVER_ID	DELIVERY_ITEM_ID
1	91011	1ZA55858541	81011	71011
2	91012	1ZA35858543	81012	71013
3	91013	1ZA17851545	81011	71015
4	91014	1ZA35868540	81013	71015
5	91015	1ZA15851545	81014	71012

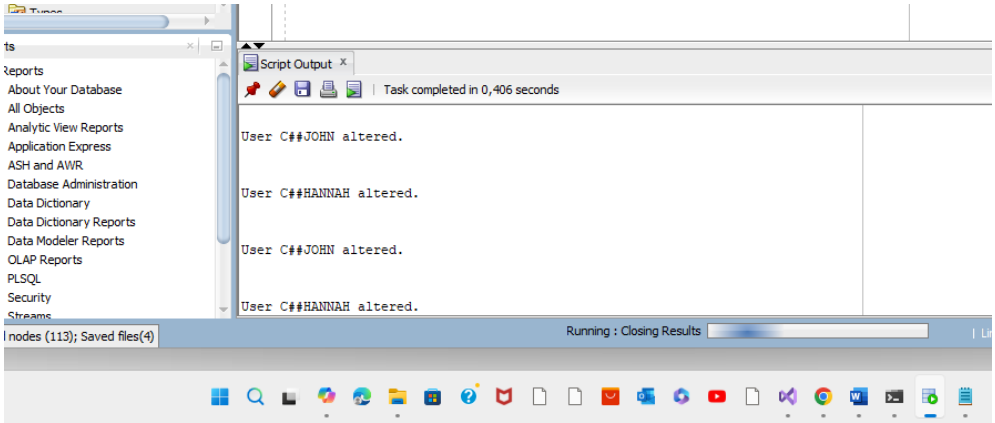
Running : Closing Results

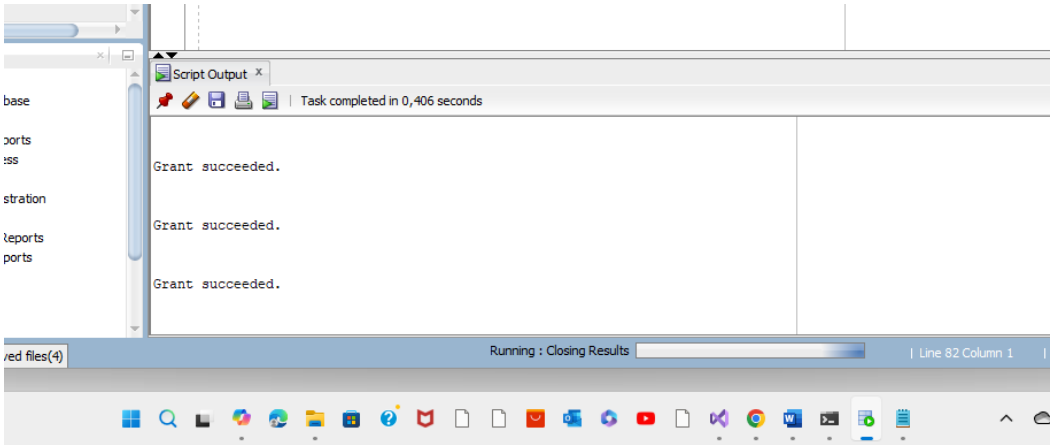
Line 20 Column 3

Question 3.1

```
-- Question 3.1

-- Create Users

CREATE USER C##John IDENTIFIED BY Johnch2024;

CREATE USER C##Hannah IDENTIFIED BY Hannah2024;


-- Set default and temporary tablespaces for both users

ALTER USER C##John DEFAULT TABLESPACE users TEMPORARY TABLESPACE temp;

ALTER USER C##Hannah DEFAULT TABLESPACE users TEMPORARY TABLESPACE temp;


-- Limit usage on both users to 100m

ALTER USER C##John QUOTA 100M ON users;

ALTER USER C##Hannah QUOTA 100M ON users;


-- Grant basic session creation privileges

GRANT CREATE SESSION TO C##John;

GRANT CREATE SESSION TO C##Hannah;


GRANT SELECT ANY TABLE TO C##John; --Allow access to John to read from any table

GRANT INSERT ANY TABLE TO C##Hannah; -- Allow access to Hannah to insert from any table
```

Output


Reports
About Your Database
All Objects
Analytic View Reports
Application Express
ASH and AWR
Database Administration
Data Dictionary
Data Dictionary Reports
Data Modeler Reports
OLAP Reports
PLSQL
Security
Script Output
Task completed in 0,406 seconds
User C##JOHN altered.
User C##HANNAH altered.
User C##JOHN altered.
User C##HANNAH altered.
Running : Closing Results


Script Output
Task completed in 0,406 seconds
Grant succeeded.
Grant succeeded.
Grant succeeded.
Running : Closing Results
Line 82 Column 1

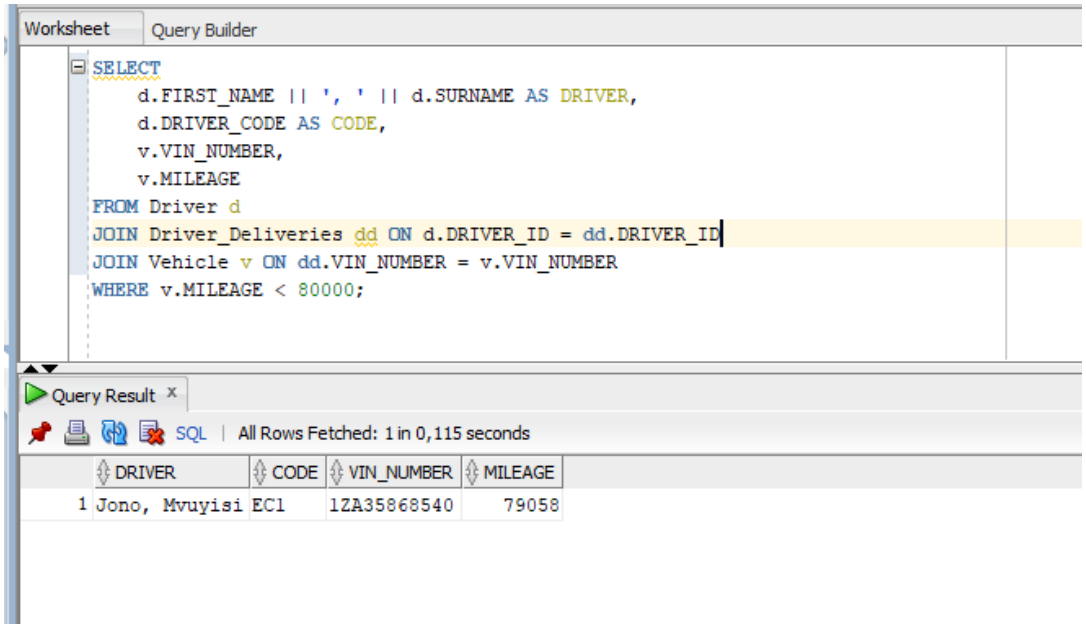
Question 3.2

The importance of separation of duties was to minimize errors by assigning different task to users. The users are granted the access that is required which is important because user John is only granted with the read only access (Microsoft,2019). He can only view data and report and audit however he cannot make changes, that provides data integrity (Microsoft,2019). With Hannah she only has access to writing data. She can add new data such as adding new customers, deliveries and billing reports but she is unable to update or make changes to data this avoids data tampering (Microsoft,2019). So, with each user granted access for specific duties this reduces fraud and errors. Cheetah deliveries ensure data integrity and security by implementing separation of duties. As it limits who can alter the company's data (Microsoft,2019).

Question 4.1

```
-- Question 4.1

-- Display vehicles with a mileage less than 80000

SELECT

   d.FIRST_NAME || ', ' || d.SURNAME AS DRIVER,

   d.DRIVER_CODE AS CODE,

   v.VIN_NUMBER,

   v.MILEAGE

FROM Driver d

JOIN Driver_Deliveries dd ON d.DRIVER_ID = dd.DRIVER_ID

JOIN Vehicle v ON dd.VIN_NUMBER = v.VIN_NUMBER

WHERE v.MILEAGE < 80000;
```

Output



Question 4.2

Relational model

Relational databases use a mechanism known as tables for keeping data in different formats or with conflicting definitions. In this case, each table has a pair of columns that are known to be key values and stored values (Educba,2023). These values are used to organize records across tables in a database system. Relational databases contain several elements, such as objects. Schemas are used to describe it (Educba,2023). For instance, one common kind of relational database is SQL, which has an ordinary interface. It promotes stability and aids in data duplication. Oracle Database Server, Microsoft SQL Server, and MySQL are a few additional relational databases (DatabaseTown,2023). When continuous document accessibility is required for data storage processes, it is typically started.

Flat File Database Model

Flat file databases store and maintain information for subsequent analysis without the requirement for an external infrastructure because they are inherently autonomous or distinct from one another. Without any further explanation beyond the file structure, they may be easily altered and produced off (Educba,2023).One file that lacks organized relationships makes up a flat file database. A data dictionary is used to describe it. For instance, the most common kind of flat file database is a CSV file. Although it is straightforward, cost effective, and straightforward to implement, there is an issue with redundant data (Educba,2023). It's not as safe. Documents, fields, data, and characters are all included. It is most frequently used in business situations where analysing an entire file is required (Educba,2023).

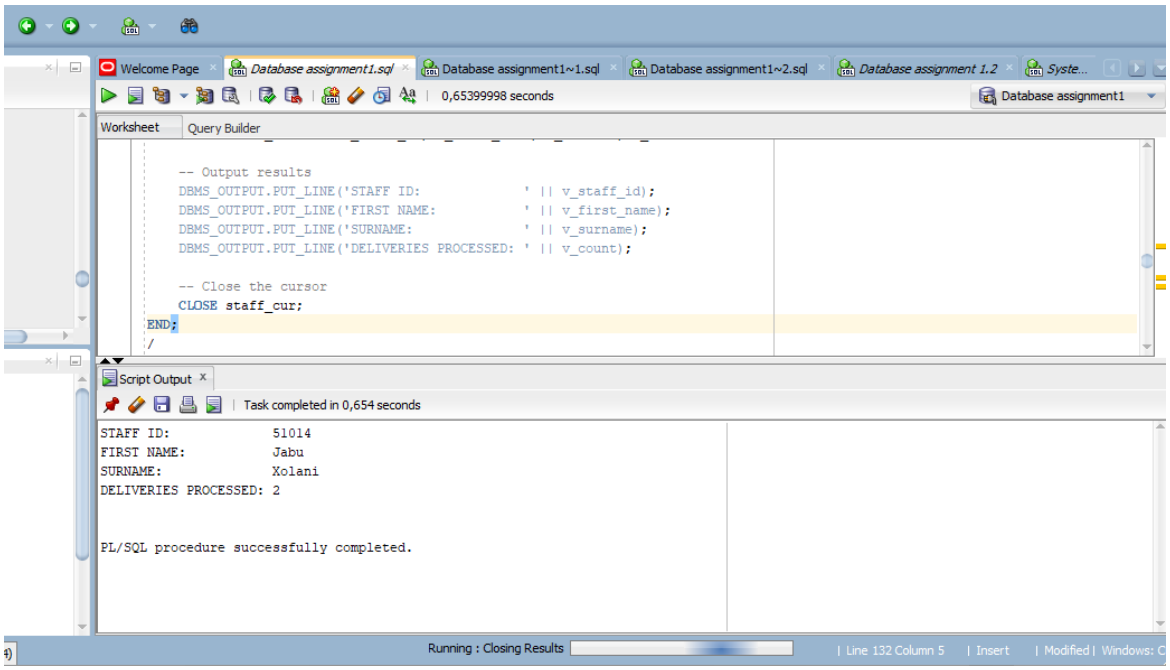
A relational model would be suitable for cheetah deliveries because it has the entity attributes and relationships and cheetah delivers have many linked entities such as customer, staff, billing, deliveries, vehicles, drivers (DatabaseTown,2023).The relational model ensures safety and security as access is limited according to tables and roles. Relational model ensures precision as they will be no incorrect or redundant customer or vehicle information. It would provide effectiveness for queries as driver, vehicle and delivery can be combined tables (DatabaseTown,2023). It would provide assistance with their objectives, which include managing billing, tracking deliveries, enhancing productivity, and keeping an eye on effectiveness (DatabaseTown,2023).

Question 5.1

```
-- Question 5.1
SET SERVEROUTPUT ON;

DECLARE
  -- Variables to hold results
  v_staff_id   Staff.Staff_ID%TYPE;
  v_first_name Staff.First_Name%TYPE;
  v_surname    Staff.Surname%TYPE;
  v_count      NUMBER;

  -- Cursor to get staff delivery counts
  CURSOR staff_cur IS
    SELECT s.Staff_ID, s.First_Name, s.Surname, COUNT(dd.Delivery_Item_ID) AS
Deliveries_Processed
    FROM Staff s
    JOIN Delivery_Item di ON s.Staff_ID = di.Staff_ID
    JOIN Driver_Deliveries dd ON di.Delivery_Item_ID = dd.Delivery_Item_ID
    GROUP BY s.Staff_ID, s.First_Name, s.Surname
    ORDER BY COUNT(dd.Delivery_Item_ID) DESC;
BEGIN
  -- Open the cursor
  OPEN staff_cur;

  -- Fetch only the first row of the staff member with most deliveries
  FETCH staff_cur INTO v_staff_id, v_first_name, v_surname, v_count;

  -- Output results
  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_count);

  -- Close the cursor
  CLOSE staff_cur;
END;
/
```

Output



Question 5.2

The PL/SQL block has three components are the declaration, execution and exception sections. The declaration section consists of the cursors, variables and constants. I declared my variables as v_staff_id and v_first_name and v_surname and I declared my cursor as staff_cur. The execution section primary logic run of the SQL operations and structures in my code I inserted a line dbms_output.put_line which would show the results after opening the cursor and fetching the highest employee (GeeksforGeeks,2024). In the Exception section its purpose is to address problems such as too many rows or data not found. Declaration made certain the variables were available in the application. The operational management staff deliverables report was produced by execution (GeeksforGeeks,2024). It would prove resilient in the event of unusual entry because of exception handling. Cheetah Deliveries enable management to provide insightful reports, including determining which employee handled the most deliveries. Setting up variables and cursors for information delivery is done in the declaration section (GeeksforGeeks,2024). To compute amounts, the execution portion runs queries via the staff, delivery item, and driver deliveries databases (GeeksforGeeks,2024). Finally, the exception section makes sure that the record functions properly even in the event of unforeseen circumstances, such no deliveries being logged during a specific time frame. Because of its architecture, PL/SQL is a useful tool for enhancing making choices, generating functional documentation, and guaranteeing the dependability of company procedures (GeeksforGeeks,2024).

Question 5.3.1

Making complicated queries simpler is one of the main goals of developing views. Accordingly, views can aid in condensing intricate queries to render them easier to read and update (Baeldung sql, 2024). In order to give users a more streamlined interface for interacting with the information, we conceal the intricacies of the actual tables and joins underneath views. Its beneficial to cheetah deliveries management because managers don't have to constantly construct intricate joins (Baeldung sql, 2024). Views improve security by employing views and displaying only a portion of the data that is accessible, access can be limited to confidential data. It is possible to conceal essential columns (Baeldung sql, 2024). By controlling computations and data modifications, views can also aid in ensuring data consistency. Cheetah deliveries could lower the possibility of errors and discrepancy that could occur when several users carry out the same changes separately when we specify such actions in a view (Baeldung sql, 2024).

Question 5.3.2

-- Question 5.3.2

-- Create a View showing all staff delivery counts

CREATE OR REPLACE VIEW StaffDeliveriesView AS

SELECT s.Staff_ID, s.First_Name, s.Surname, COUNT(dd.Delivery_Item_ID) AS Deliveries_Processed

FROM Staff s

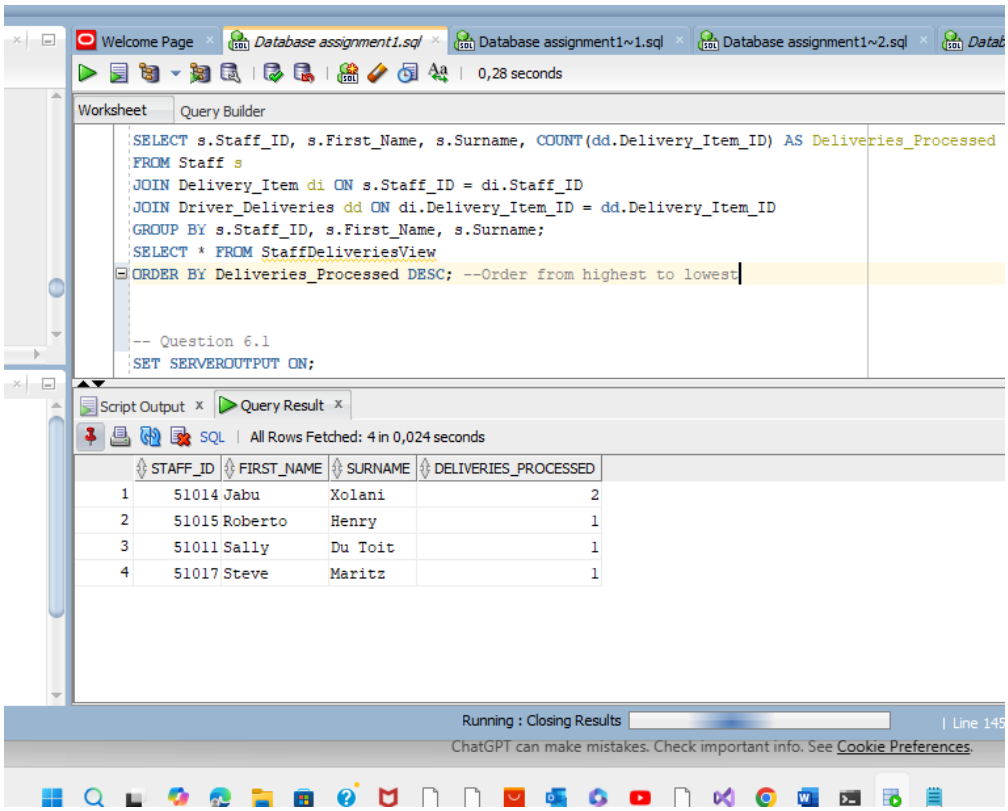
JOIN Delivery_Item di ON s.Staff_ID = di.Staff_ID

JOIN Driver_Deliveries dd ON di.Delivery_Item_ID = dd.Delivery_Item_ID

GROUP BY s.Staff_ID, s.First_Name, s.Surname;

SELECT * FROM StaffDeliveriesView ORDER BY Deliveries_Processed DESC; --Order from highest to lowest

Output



Question 6.1

A. Implicit cursors

An implicit cursor that contains the rows impacted by the specific DML activity is produced whenever any DML activities take place in the database (Guru99, 2024). Since these cursors are nameless, it is impossible to regulate or refer to them from another section of the code (Guru99, 2024). The cursor properties only allow us to make reference to the most current cursor. Implicit cursor attributes are helpful in Cheetah Deliveries when managers or employees perform one-row tasks like removing a discontinued billing record or adjusting vehicle mileage following a delivery. For instance, the business can use SQL%ROWCOUNT to quickly verify what number of rows were impacted after changing a vehicle's mileage (Guru99, 2024). This eliminates the need to set up a unique cursor and guarantees that modifications were effective and that the confidentiality of information is preserved (Guru99, 2024). Attributes like:

%ISOPEN:The outcome of sql%isopen always equals false (Oracle,2025).
%FOUND: One of the following values is present in sql%found: null if neither a dml nor select statement has executed. True if a row was returned by the most relevant DML or select statement. false if no row was returned by the most current select or DML operation (Oracle,2025).
%Not Found: One of these values is present in sql%notfound: null if neither a DML nor select statement has executed. false if a row was returned by the most current DML or select operation. True if no row was returned by the most current select or DML operation (Oracle,2025).
%ROWCOUNT: One of these numbers is present in sql%rowcount: null if neither a DML nor select statement has executed. The number of rows that have been fetched thus far, if a select or DML statement has executed. (Oracle,2025).

B. Explicit Cursors

To have more authority over their DML activities, developers can construct a designated setting area (Guru99, 2024). The explicit cursor is made for the "SELECT" statement that must be utilized in the code and must be specified in the PL/SQL block's declaration section. Declaring the cursor Creating a unique reference area for the select statement specified by the declaration section is all that is required to declare the cursor (Guru99, 2024). The cursor and this semantic area have a single name. When management need to create multiple row results in Cheetah Deliveries, such identifying all drivers and the cars they are allocated or figuring out how many deliveries each driver has accomplished, explicit cursor properties are useful (Guru99, 2024). Explicit cursors enable managers to monitor service quality driver by driver by iterating across several records. For instance, employing driver_cur%ROWCOUNT during looping enables

management to track scheduling and performance by displaying the precise number of delivery entries handled (Guru99, 2024). Attributes include:

Cursor_name%FOUND-TRUE, if at least one record is returned by the fetch statement. false, if no row is returned by the get statement (PL/SQL tutorial,2025).

Cursor_name%NOTFOUND-TRUE, if a row is not returned by the get operation. If the retrieve statement yields at least one record false (PL/SQL tutorial,2025).

Cursor_name%ROWCOUNT- is the number of rows that the fetch statement retrieved. The PL/SQL command returns an error if no row is returned (PL/SQL tutorial,2025).

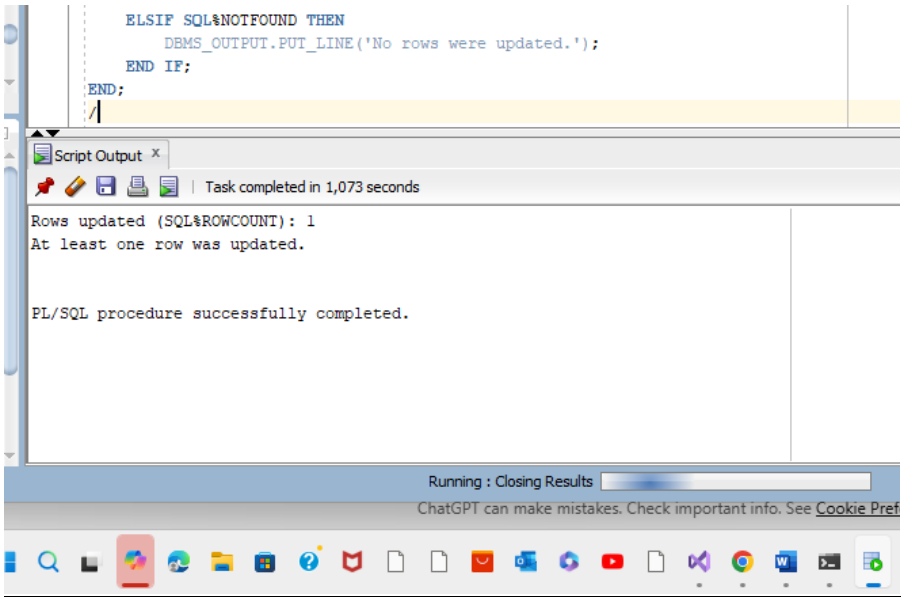
Cursor_name%ISNAME- true, provided that the application's has been previously open. false, if the application does not have the cursor open (PL/SQL tutorial,2025).

Question 6.1 A- Implicit Cursor Example

```
-- Question 6.1 A- Implicit Cursor

SET SERVEROUTPUT ON;


BEGIN

  -- Example of implicit cursor attributes

  UPDATE Vehicle

  SET Mileage = Mileage + 1000

  WHERE VIN_Number = '1ZA35868540';


  -- SQL%ROWCOUNT gives the number of rows affected by the UPDATE

  DBMS_OUTPUT.PUT_LINE('Rows updated (SQL%ROWCOUNT): ' || SQL%ROWCOUNT);


  -- SQL%FOUND checks if at least one row was affected

  IF SQL%FOUND THEN

    DBMS_OUTPUT.PUT_LINE('At least one row was updated.');

  -- SQL%NOTFOUND checks if no rows were affected

  ELSIF SQL%NOTFOUND THEN

    DBMS_OUTPUT.PUT_LINE('No rows were updated.');
```

```
  END IF;
END;
/
```

Output

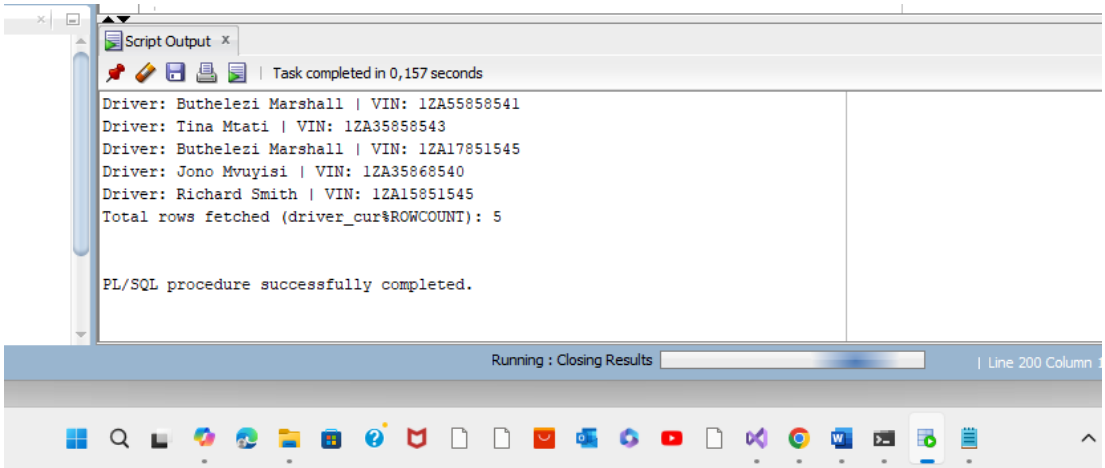


Question 6.1 B- Explicit Cursor Example

```
-- Question 6.1 B- Explicit Cursor
SET SERVEROUTPUT ON;

DECLARE
  -- Explicit cursor declaration: fetches drivers and the vehicles they used
  CURSOR driver_cur IS
    SELECT d.First_Name, d.Surname, dd.VIN_Number
    FROM Driver d
    JOIN Driver_Deliveries dd ON d.Driver_ID = dd.Driver_ID;

  -- Variables to store the fetched row values
  v_first   Driver.First_Name%TYPE;
  v_surname Driver.Surname%TYPE;
```

```
  v_vin    Driver_Deliveries.VIN_Number%TYPE;
BEGIN
  -- Open the cursor to start fetching data
  OPEN driver_cur;
  DBMS_OUTPUT.PUT_LINE('Driver delivery records:');

  -- Loop through the result set one row at a time
  LOOP
    FETCH driver_cur INTO v_first, v_surname, v_vin;

    -- Exit the loop when no more rows are found
    EXIT WHEN driver_cur%NOTFOUND;

    -- Display the fetched row
    DBMS_OUTPUT.PUT_LINE('Driver: ' || v_first || ' ' || v_surname ||
                    ' | VIN: ' || v_vin);
  END LOOP;

  -- Cursor attribute %ROWCOUNT shows how many rows were fetched in total
  DBMS_OUTPUT.PUT_LINE('Total rows fetched: ' || driver_cur%ROWCOUNT);

  -- Close the cursor to release memory
  CLOSE driver_cur;
END;
/
```

Output



Question 6.2

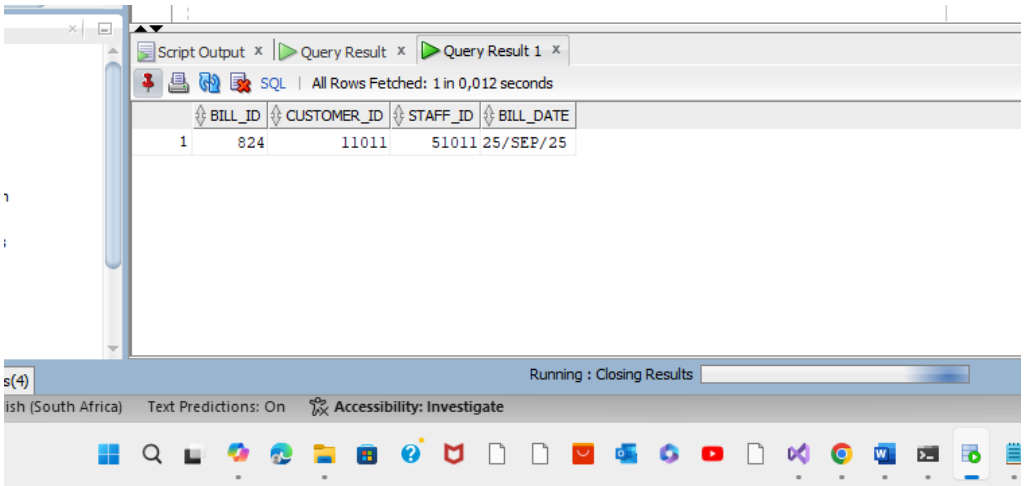
Configurable database objects called SQL sequences are made to produce a string of numbers (GeeksforGeeks, 2025). Sequences are autonomous objects that can be utilized across various tables, in contrast to identification columns, which are closely tied to particular tables. Sequences can be used as main or distinct keys in table formats since they offer distinctive values (GeeksforGeeks, 2025). It is possible to set up sequences to produce values in either a descending or a rising order (GeeksforGeeks, 2025). Sequences are autonomous and applicable to various tables, in contrast to identification columns. Sequences save hours of software code by lowering the administrative burden and difficulty involved in manually creating unique variables (GeeksforGeeks, 2025). Cheetah Deliveries have to use sequence because every billing record needs to have its own bill id. Managers can avoid manually assigning id's by employing a sequence, which ensures integrity in the database (GeeksforGeeks, 2025).

Question 6.2- Code

```
-- Question 6.2

-- Create the sequence for billing IDs

CREATE SEQUENCE bill_seq

START WITH 821  -- starting value

INCREMENT BY 1

MINVALUE 1  -- Min value
```

```
MAXVALUE 9999  -- Max value
NOCYCLE
NOCACHE;   -- Specifies that no value in the sequence is cached in memory

-- Insert into Billing using NEXTVAL
INSERT INTO Billing (Bill_ID, Customer_ID, Staff_ID, Bill_Date)
VALUES (bill_seq.NEXTVAL, 11011, 51011, SYSDATE);

-- Show the current sequence value
SELECT bill_seq.CURRVAL AS Last_Used_ID FROM dual;

-- Verify the new row in Billing
SELECT *
FROM Billing
ORDER BY Bill_ID DESC   -- Descending order
FETCH FIRST 1 ROWS ONLY;
```

Output



References

Araromi.A.Baeldung sql. 2024.Understanding the Purpose of Creating a View in a Database, 13 September 2024

[Online]. Available at: https://www.baeldung.com/sql/database-view-purpose

[Accessed 23 September 2025]

Brown.F. Guru99. 2024.Oracle PL/SQL Cursor: Implicit, Explicit, For Loop with Example,28 June 2024

[Online]. Available at: https://www.guru99.com/pl-sql-cursor.html

[Accessed 23 September 2025]

Educba. 2023.Relational database vs Flat file, 13 March 2023

[Online]. Available at: https://www.educba.com/relational-database-vs-flat-file/

[Accessed 23 September 2025]

DatabaseTown. 2023.Difference Between Flat File VS Database, 31 January 2023

[Online]. Available at: https://databasetown.com/difference-between-flat-file-vs-database/

[Accessed 23 September 2025]

GeeksforGeeks. 2024.Cursors in PL/SQL, 17 May 2024

[Online]. Available at: https://www.geeksforgeeks.org/sql/cursors-in-pl-sql/

[Accessed 23 September 2025]

GeeksforGeeks.2025.SQL Sequence, 13 January 2025

[Online]. Available at: https://www.geeksforgeeks.org/sql/sql-sequences/

[Accessed 23 September 2025]

Microsoft.Tech Community. 2019.Seperation of duties for DBA's, 23 March 2019

[Online]. Available at: https://techcommunity.microsoft.com/blog/sqlserver/separation-of-duties-for-dbas/383915

[Accessed 23 September 2025]

Oracle. 2025.Database Pl/SQL language reference implicit cursors, 2025

[Online]. Available at: https://docs.oracle.com/en/database/oracle/oracle-database/21/lnpls/implicit-cursor-attribute.html

[Accessed 23 September 2025]

PL/SQL tutorial. 2025.Explicit Cursors, 2025

[Online]. Available at: https://plsql-tutorial.com/plsql-explicit-cursors.htm

[Accessed 23 September 2025]