ST10436137

Dario Hendrickse

INSY7213

Assignment 1

Table of Contents

[Question 1 3](#_Toc209821059)

[ERD 3](#_Toc209821060)

[Assumptions 3](#_Toc209821061)

[Question 2 5](#_Toc209821062)

[Database Schema and Tables 5](#_Toc209821063)

[Data Values Inserted into Tables 6](#_Toc209821064)

[Customers Table 7](#_Toc209821065)

[Staff Table 8](#_Toc209821066)

[Driver Table 9](#_Toc209821067)

[Vehicle Table 10](#_Toc209821068)

[Delivery\_Items Table 11](#_Toc209821069)

[Billing Table 12](#_Toc209821070)

[Driver\_Deliveries Table 13](#_Toc209821071)

[Question 3 14](#_Toc209821072)

[Q3.1: Creating User John and Hannah and Granting Privileges 14](#_Toc209821073)

[SQL Code: 14](#_Toc209821074)

[Screenshot: 15](#_Toc209821075)

[Successful Implementation 15](#_Toc209821076)

[Q3.2: Importance of Segregation of Duties 16](#_Toc209821077)

[Question 4 17](#_Toc209821078)

[Q4.1: PL/SQL Query 17](#_Toc209821079)

[SQL Code 17](#_Toc209821080)

[Screenshot of SQL Code 18](#_Toc209821081)

[Screenshot of Successful Output 19](#_Toc209821082)

[Q4.2: Flat File Database vs Relational Database 19](#_Toc209821083)

[Question 5 20](#_Toc209821084)

[Q5.1: PL/SQL Query 20](#_Toc209821085)

[SQL Code 20](#_Toc209821086)

[Screenshot of SQL Code 21](#_Toc209821087)

[Successful Output Screenshot 22](#_Toc209821088)

[Q5.2: PL/SQL Components 22](#_Toc209821089)

[Q5.3.1: View Explanation 23](#_Toc209821090)

[Q5.3.2: SQL Code for the View 23](#_Toc209821091)

[SQL Code 23](#_Toc209821092)

[Screenshot of SQL Code 24](#_Toc209821093)

[Screenshot of Successful Output 24](#_Toc209821094)

[Question 6 25](#_Toc209821095)

[Q6.1: Implicit and Explicit Cursors 25](#_Toc209821096)

[Motivations for Implicit Cursors 25](#_Toc209821097)

[Motivation for Explicit Cursors 25](#_Toc209821098)

[Implicit Cursor Code 26](#_Toc209821099)

[Explicit Cursor Code 26](#_Toc209821100)

[Screenshot of Code in SQL Developer 28](#_Toc209821101)

[Screenshot of Successful Cursor Outputs 29](#_Toc209821102)

[Q6.2: Using a Sequence 29](#_Toc209821103)

[SQL Code 29](#_Toc209821104)

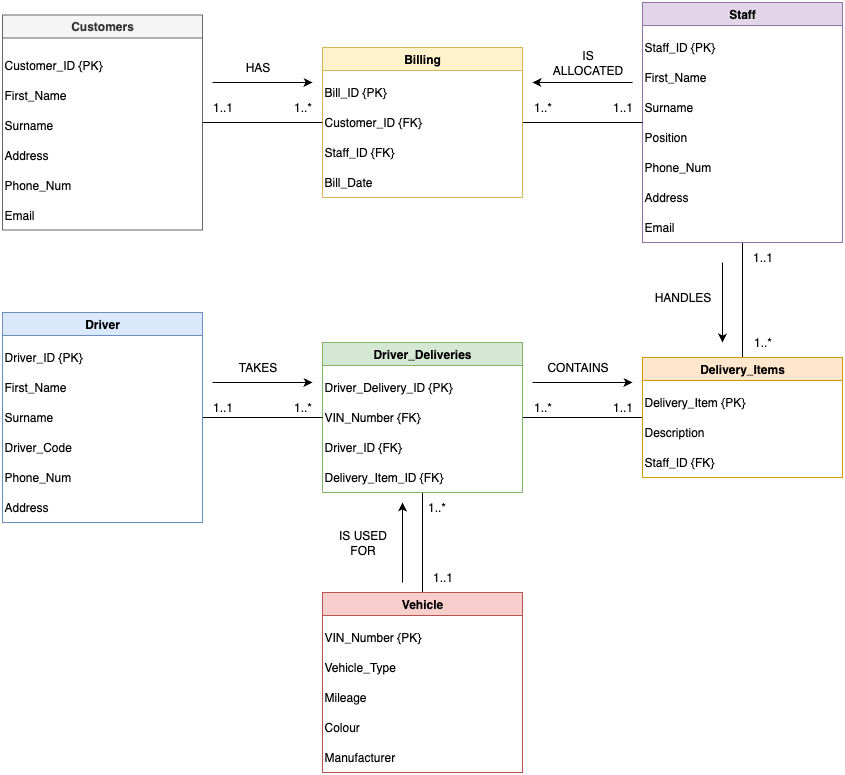
[Screenshot of Code 30](#_Toc209821105)

[Screenshot of Successful Implementation 30](#_Toc209821106)

[References 31](#_Toc209821107)

# Question 1

## ERD



## Assumptions

* All attributes in the flat files are the only attributes in each table.
* Each billing is for one customer only, but a customer can have multiple billings.
* Each billing is allocated to one staff member, however, each staff member can have multiple billings.
* One staff member can handle various delivery items, however, there is only one staff member per delivery item.
* Driver deliveries can only have one vehicle, driver, and delivery item per instance.
* Each vehicle can go on multiple deliveries.
* Each driver can perform multiple deliveries.
* Each delivery item can be used for multiple deliveries.

# Question 2

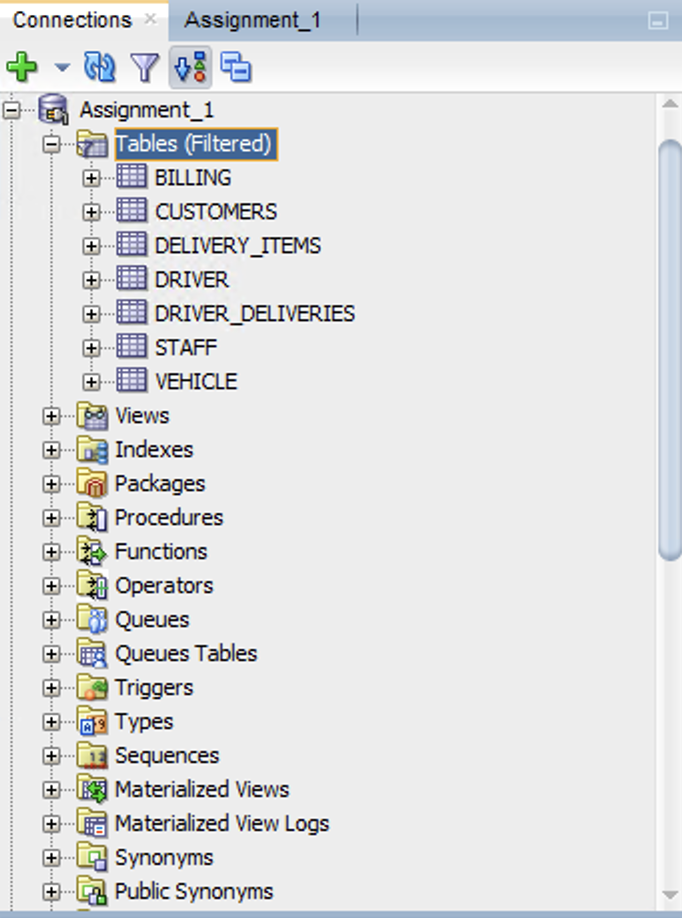
All tables were created using the .csv files provided in Student Materials. All data was also inserted in this manner.

All tables were queried to view all data to ensure the process was done correctly.

## Database Schema and Tables

A new connection named Assignment\_1 was created and all tables were created under this connection.

**Screenshot of the above:**



## Data Values Inserted into Tables

All table data was queried to show that all data was inserted successfully.

**The SQL Code for all queries was the following:**

SELECT \* FROM Customers;

SELECT \* FROM Staff;

SELECT \* FROM Driver;

SELECT \* FROM Vehicle;

SELECT \* FROM Billing;

SELECT \* FROM Delivery\_Items;

SELECT \* FROM Driver\_Deliveries;

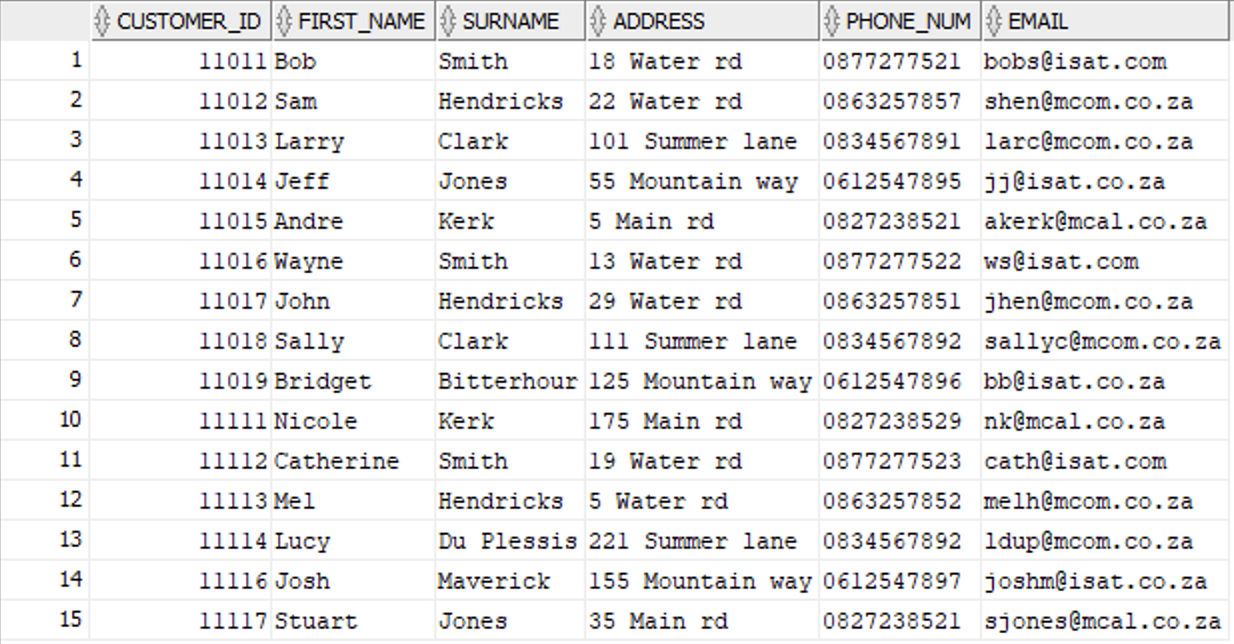
### Customers Table

.csv import:

A screenshot of a computer

AI-generated content may be incorrect.

Table data:



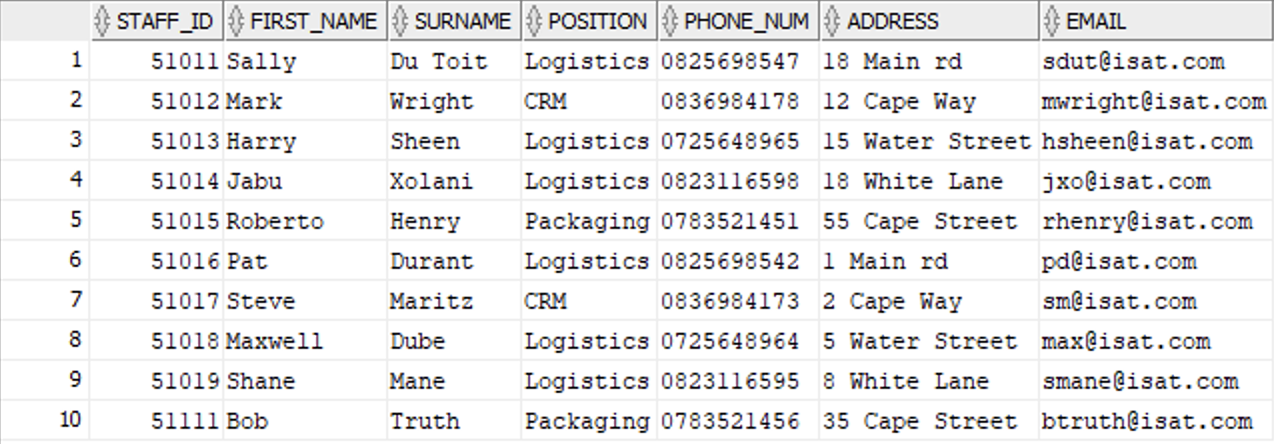
### Staff Table

.csv import:

A screenshot of a computer

AI-generated content may be incorrect.

Table data:



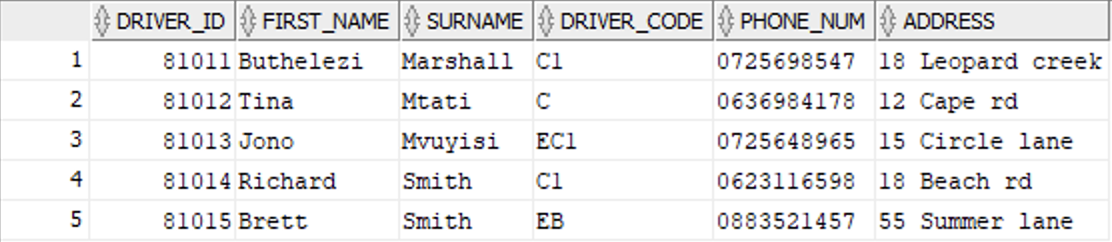
### Driver Table

.csv import:

A screenshot of a computer

AI-generated content may be incorrect.

Table data:



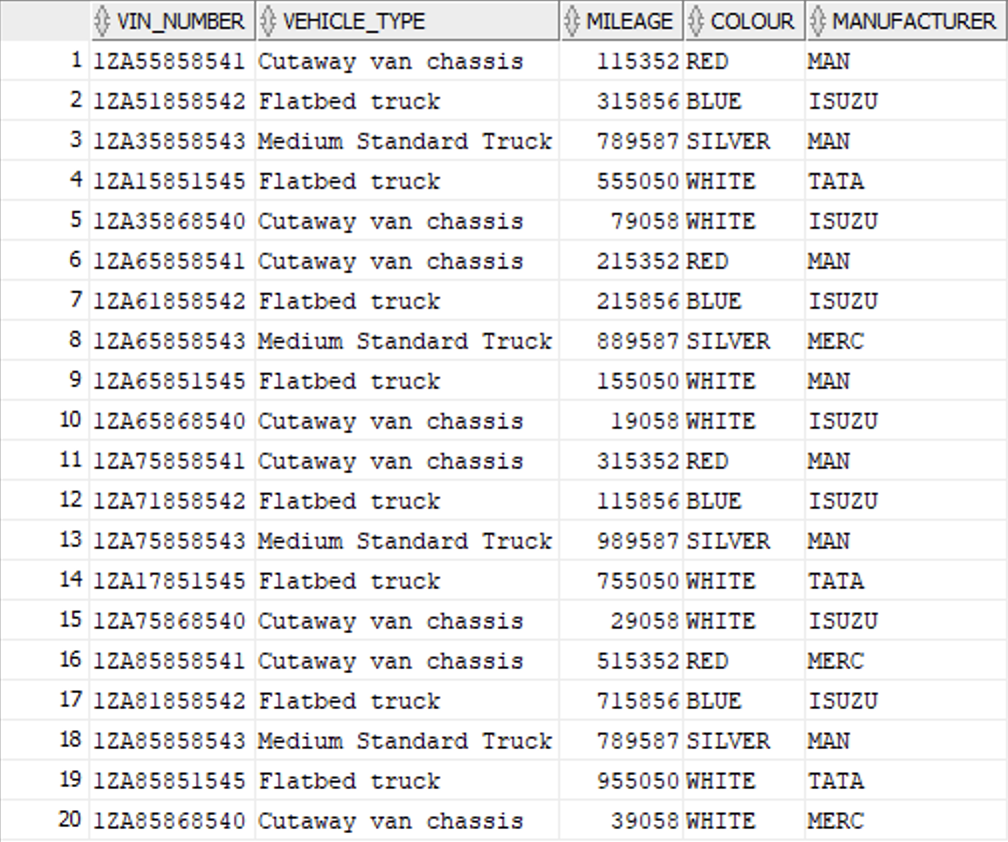
### Vehicle Table

.csv import:

A screenshot of a computer

AI-generated content may be incorrect.

Table data:



### Delivery\_Items Table

.csv import:

A screenshot of a computer

AI-generated content may be incorrect.

Table data:



### Billing Table

.csv import:

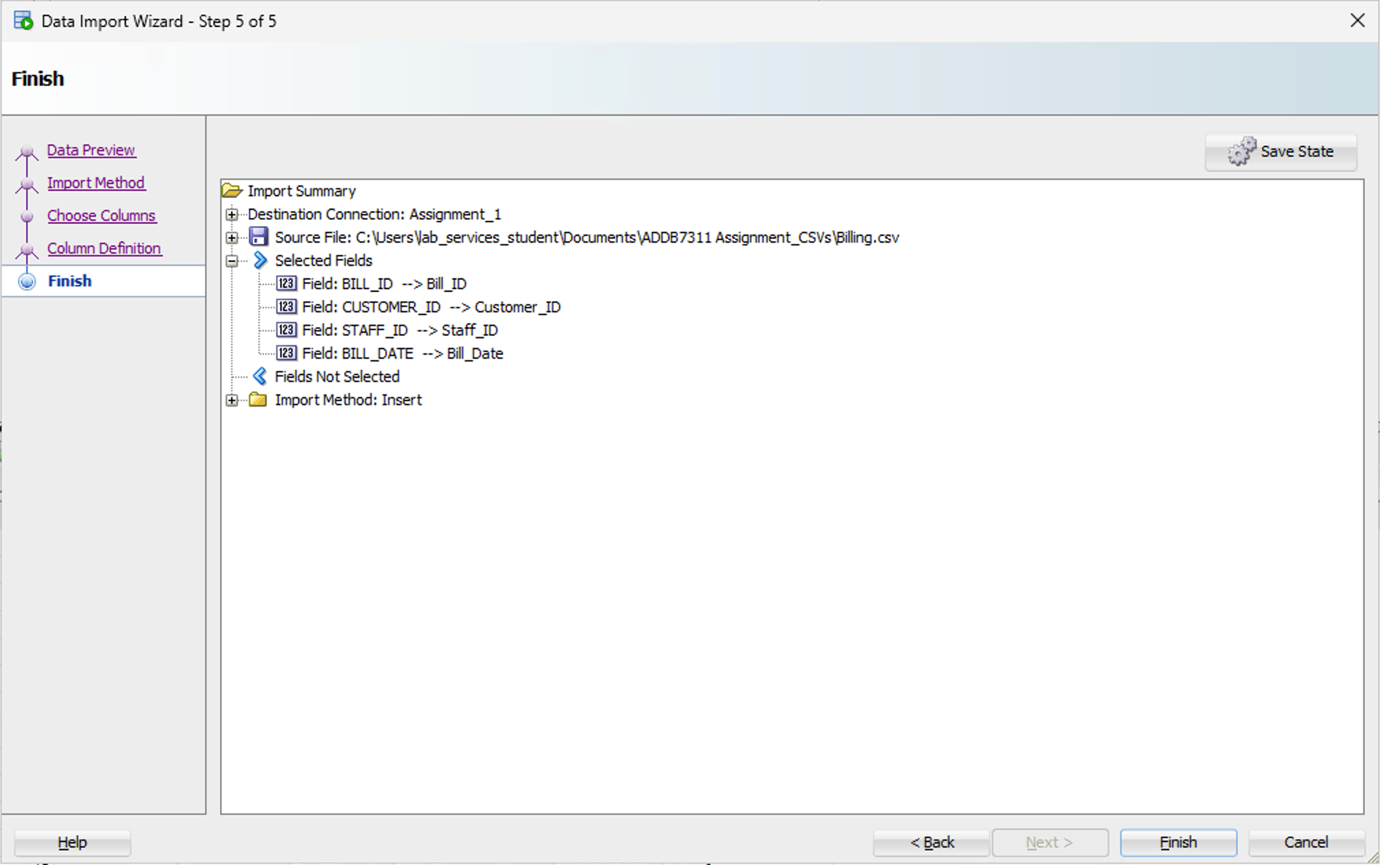
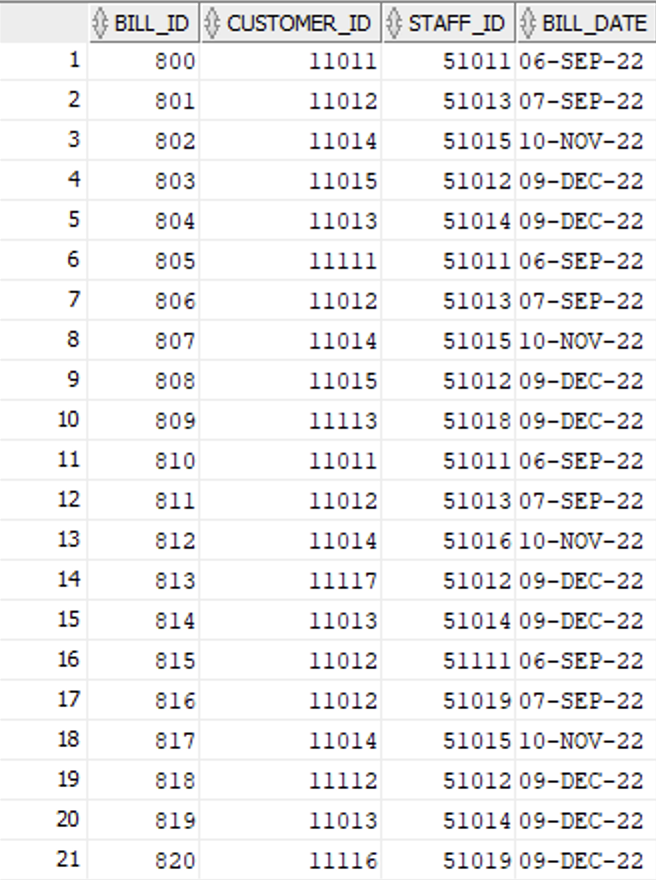


Table data:



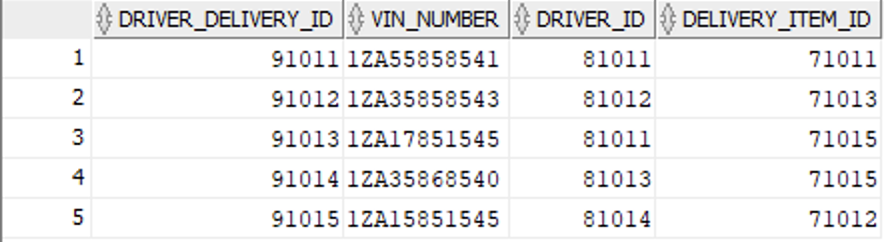
### Driver\_Deliveries Table

.csv import:

A screenshot of a computer

AI-generated content may be incorrect.

Table data:



# Question 3

The creation of users and granting them privileges was sourced from Oracle (2018).

## Q3.1: Creating User John and Hannah and Granting Privileges

### SQL Code:

-- Setting the correct container

ALTER SESSION SET CONTAINER="XEPDB1";

-- Creating User John

CREATE USER John IDENTIFIED BY "Johnch2024";

--Creating User Hannah

CREATE USER Hannah IDENTIFIED BY "Hannahch2024";

-- Granting SELECT ANY TABLE privilege to John

GRANT CREATE SESSION TO John;

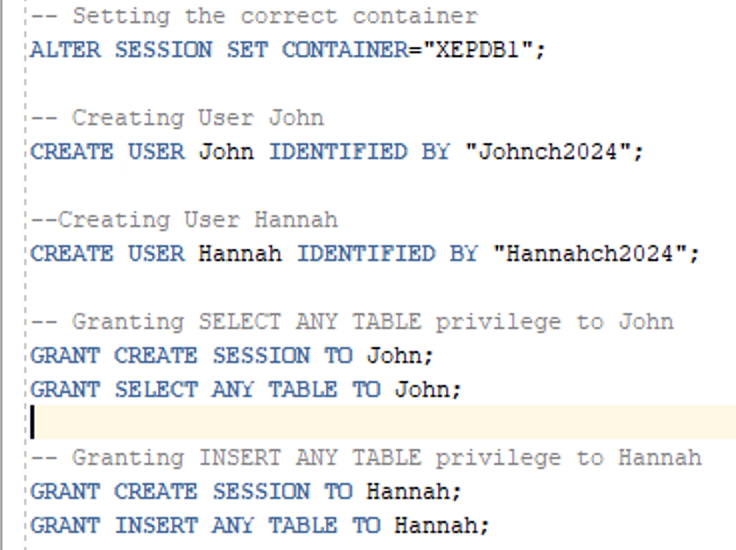
GRANT SELECT ANY TABLE TO John;

-- Granting INSERT ANY TABLE privilege to Hannah

GRANT CREATE SESSION TO Hannah;

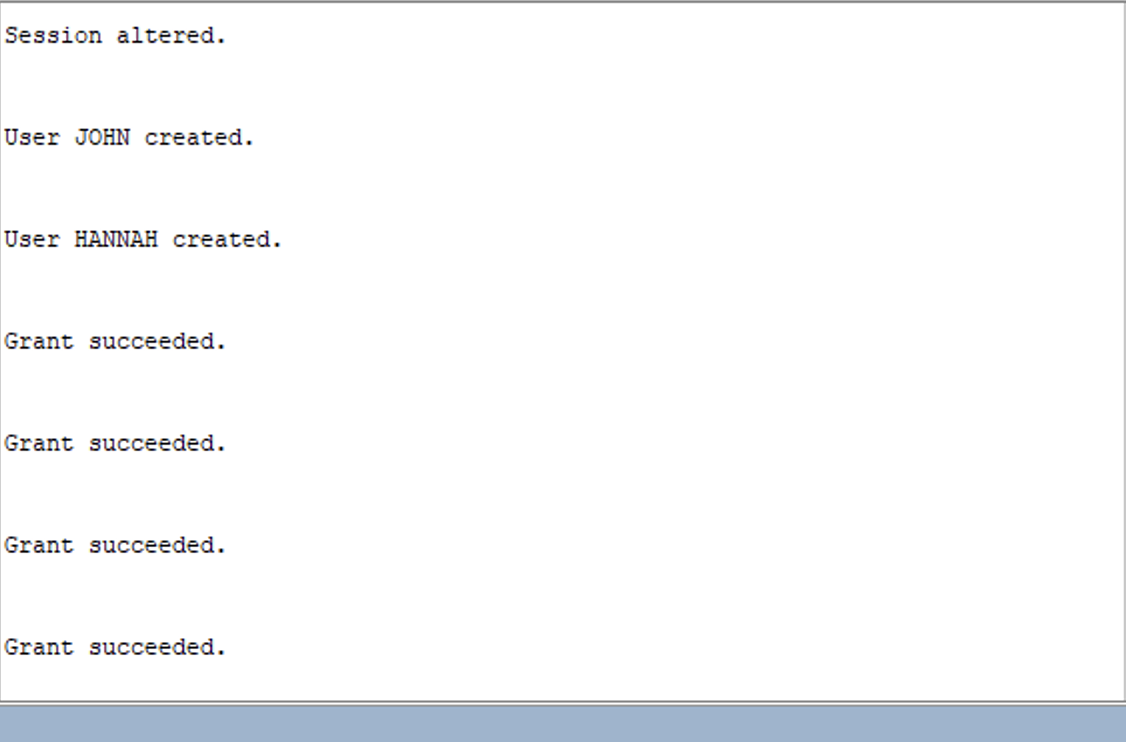
GRANT INSERT ANY TABLE TO Hannah;

### Screenshot:



## Successful Implementation

Screenshot showing that all actions were performed successfully:



## Q3.2: Importance of Segregation of Duties

Segregation of Duties is “an internal control mechanism designed to prevent errors and fraud by ensuring at least two individuals are responsible for the separate parts of any task” (Hanna, 2025). Therefore, in the context of Cheetah Deliveries, separating the duties for selecting data and inserting data will help minimize risk of errors. Providing John and Hannah with their respective responsibilities means they have allows them to invest more effort into their tasks as there is less for them to do. Moreover, this will help reduce the risk of errors when they perform their tasks, meaning that overall the efficiency of the business’ database operations is maximised. Furthermore, entrusting separate duties to different individuals means that the business knows who is responsible should something go wrong. In the context of Cheetah Deliveries, if an instance is inserted incorrectly, then the business is aware that this was Hannah’s mistake as her only privilege is to insert data.

# Question 4

## Q4.1: PL/SQL Query

### SQL Code

SET SERVEROUTPUT ON;

DECLARE

CURSOR curs\_driver IS

SELECT d.First\_name, d.Surname, d.Driver\_Code, dd.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;

BEGIN

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

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

for r in curs\_driver LOOP

DBMS\_OUTPUT.PUT\_LINE('DRIVER: ' || r.First\_Name || ', ' || r.Surname);

DBMS\_OUTPUT.PUT\_LINE('CODE: ' || r.Driver\_Code);

DBMS\_OUTPUT.PUT\_LINE('VIN NUMBER: ' || r.VIN\_Number);

DBMS\_OUTPUT.PUT\_LINE('Mileage: ' || r.Mileage);

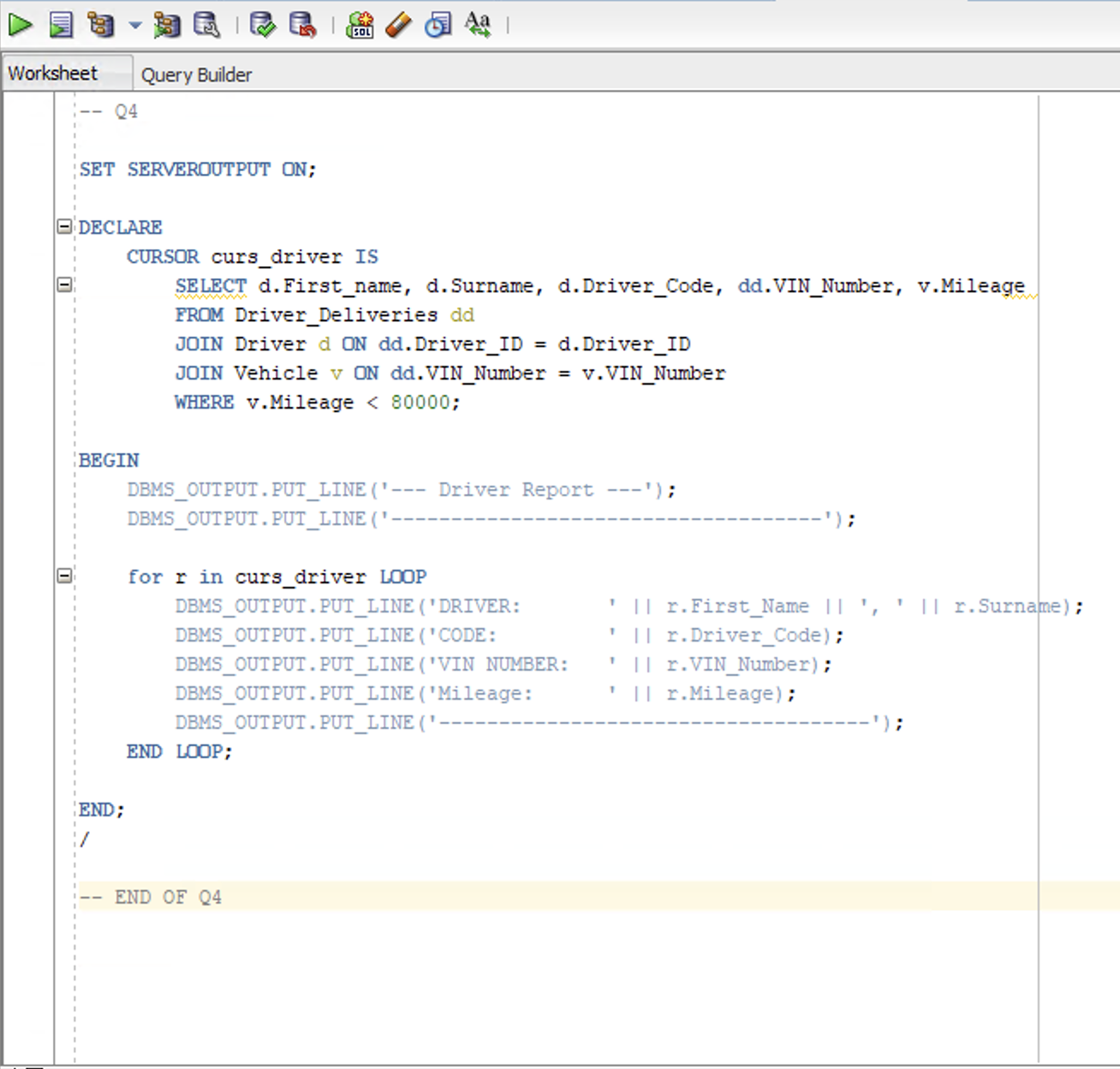
DBMS\_OUTPUT.PUT\_LINE('------------------------------------');

END LOOP;

END;

/

### Screenshot of SQL Code



### Screenshot of Successful Output

A screenshot of a computer

AI-generated content may be incorrect.

## Q4.2: Flat File Database vs Relational Database

A Flat File Database Model “stores data in plain text format and is organized as a single table with no relationships between tables” (DatabaseTown, 2023). Therefore, each table has no primary or foreign keys and each of these exists in separate files, allowing for no relation between tables.

A Relational Database Model is “a type of database that stores and organises data in a collection of tables” (DatabaseTown, 2023). DatabaseTown (2023) further mentions that these database models use a common field called a primary key in each table as a method of relating multiple tables to each other. These databases store, organise and retrieve data very efficiently and allow for quick database operations (DatabaseTown, 2023).

Therefore, Cheetah Deliveries would be best suited to using the relational database model. This is because each table in their database has a common field which can be used for a primary key. Furthermore, there are relationships between different tables, meaning that using these primary key values as foreign keys in these tables is beneficial and well-suited. An example of a table that would make use of this is Billing; there is a relationship to both Customers and Staff. Also, a relational database model means that operations can be performed quicker due to the inherent organisation and retrieval capabilities, meaning that they could improve on their delivering packages behind schedule, allowing their clientele to be more satisfied.

# Question 5

## Q5.1: PL/SQL Query

### SQL Code

-- Q5.1

SET SERVEROUTPUT ON;

DECLARE

CURSOR curs\_top\_staff IS

SELECT

s.Staff\_ID AS staff\_id,

s.First\_Name AS first\_name,

s.Surname AS surname,

COUNT(s.Staff\_ID) AS deliveries\_processed

FROM Staff s

JOIN Delivery\_Items di ON di.Staff\_ID = s.Staff\_ID

JOIN Driver\_Deliveries dd ON dd.Delivery\_Item\_ID = di.Delivery\_Item

GROUP BY s.Staff\_ID, s.First\_Name, s.Surname

ORDER BY COUNT(s.Staff\_ID) DESC

FETCH FIRST 1 ROWS ONLY;

BEGIN

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

FOR r IN curs\_top\_staff LOOP

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

DBMS\_OUTPUT.PUT\_LINE('STAFF ID: ' || r.staff\_id);

DBMS\_OUTPUT.PUT\_LINE('FIRST NAME: ' || r.first\_name);

DBMS\_OUTPUT.PUT\_LINE('SURNAME: ' || r.surname);

DBMS\_OUTPUT.PUT\_LINE('DELIVERIES PROCESSED: ' || r.deliveries\_processed);

END LOOP;

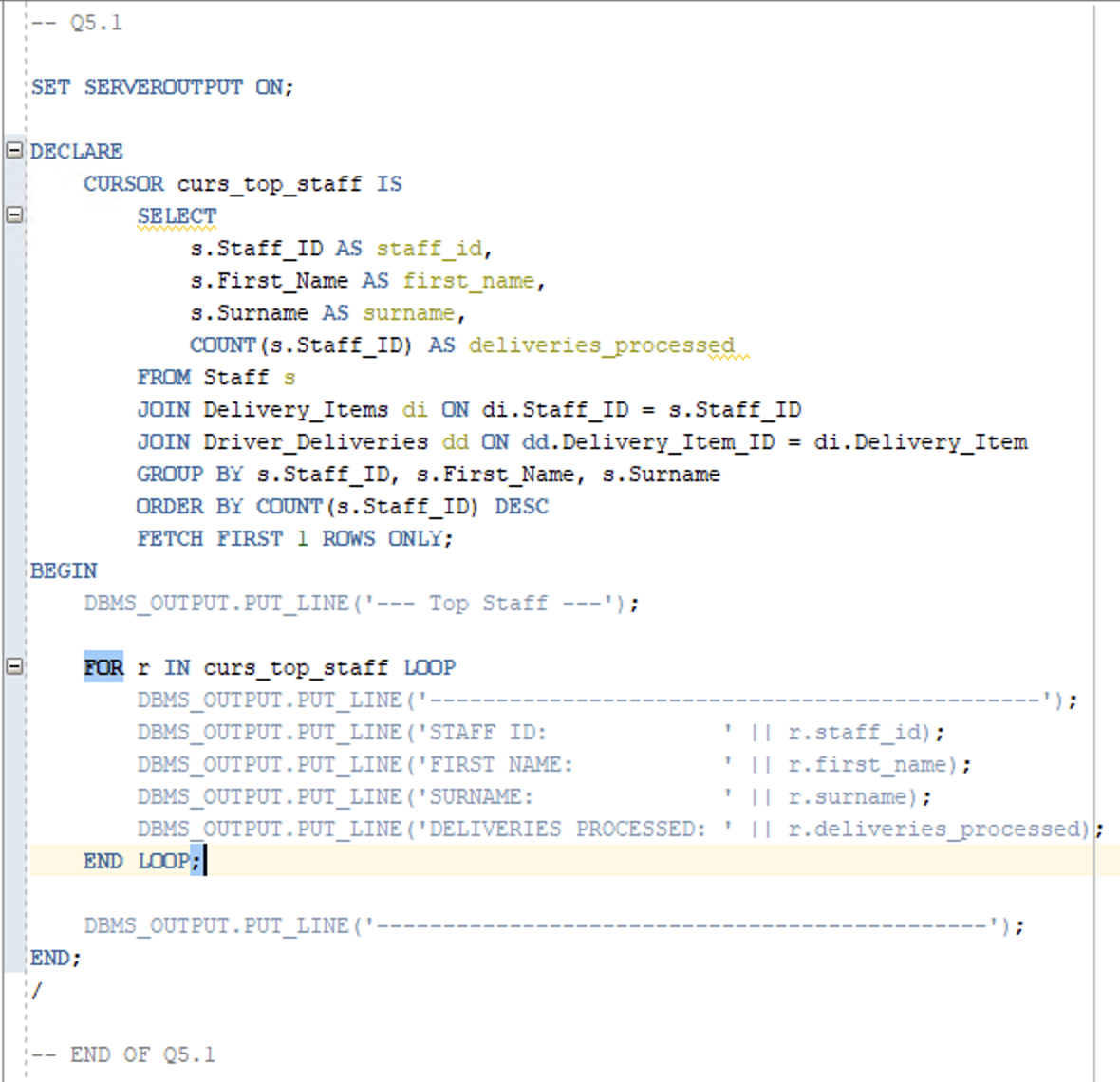
DBMS\_OUTPUT.PUT\_LINE('----------------------------------------------');

END;

/

-- END OF Q5.1

### Screenshot of SQL Code



### Successful Output Screenshot

A close-up of a document

AI-generated content may be incorrect.

## Q5.2: PL/SQL Components

PL/SQL queries consist of three parts: declarative part, executable part, and exception-handling part (Rosenzweig and Rakhimov, 2023).

Rosenzweig and Rakhimov (2023) mentions that the declarative part is optional and not necessary for PL/SQL queries. It is the first section and in this section, variables are declared that will be used in the execution of the query (Rosenzweig and Rakhimov, 2023). This section is identified by the keyword ‘DECLARE’ (Rosenzweig and Rakhimov, 2023).

Rosenzweig and Rakhimov (2023) states that the executable part is mandatory for PL/SQL queries. Furthermore, in this section of the query, the code is executed and processes an output for the console (Rosenzweig and Rakhimov, 2023). This is the second section of the code and is identified by the keyword ‘BEGIN’ (Rosenzweig and Rakhimov, 2023).

Rosenzweig and Rakhimov (2023) mentions that the exception-handling section is not mandatory. Furthermore, this section is the last section of the code and will catch any mentioned errors that are declared in this section (Rosenzweig and Rakhimov, 2023). This section is identified by the keyword ‘EXCEPTION’ (Rosenzweig and Rakhimov, 2023).

Therefore, only the declarative and executable sections were used in the PL/SQL query for Q5.1. The declarative section involved declaring the cursor curs\_top\_staff which will be used in the following executable section. The executable section then uses the declared cursor and compiles an output for the console and then outputs this. As mentioned, the Q5.1 query does not contain an exception-handling section.

## Q5.3.1: View Explanation

A view is “a saved SQL query that acts as a virtual table” (GeeksforGeeks, 2025). GeeksforGeeks (2025) goes on to mention that views do not save data like normal tables but instead call the saved query each time to access the data.

One benefit is that this allows for simplified complex queries by “encapsulating complex joins and conditions into a single object” (GeeksforGeeks, 2025). This makes life easier for developers as they don’t have to write out an SQL query each time they wish to access this data.

Another benefit is that the data presented is flexible, meaning that different views can be created for different relevant users (GeeksforGeeks, 2025). This is beneficial to Cheetah Deliveries as we have shown that we would like to segregate duties amongst employees, meaning that we can create different views for the respective users.

## Q5.3.2: SQL Code for the View

### SQL Code

-- Q5.3.2

CREATE OR REPLACE VIEW View\_TopStaff AS -- Creating the view with the desired name (virtual table)

-- Select query that will be saved to the view and be used every time its accessed

SELECT

s.Staff\_ID AS staff\_id,

s.First\_Name AS first\_name,

s.Surname AS surname,

COUNT(s.Staff\_ID) AS deliveries\_processed

FROM Staff s

JOIN Delivery\_Items di ON di.Staff\_ID = s.Staff\_ID

JOIN Driver\_Deliveries dd ON dd.Delivery\_Item\_ID = di.Delivery\_Item

GROUP BY s.Staff\_ID, s.First\_Name, s.Surname

ORDER BY COUNT(s.Staff\_ID) DESC

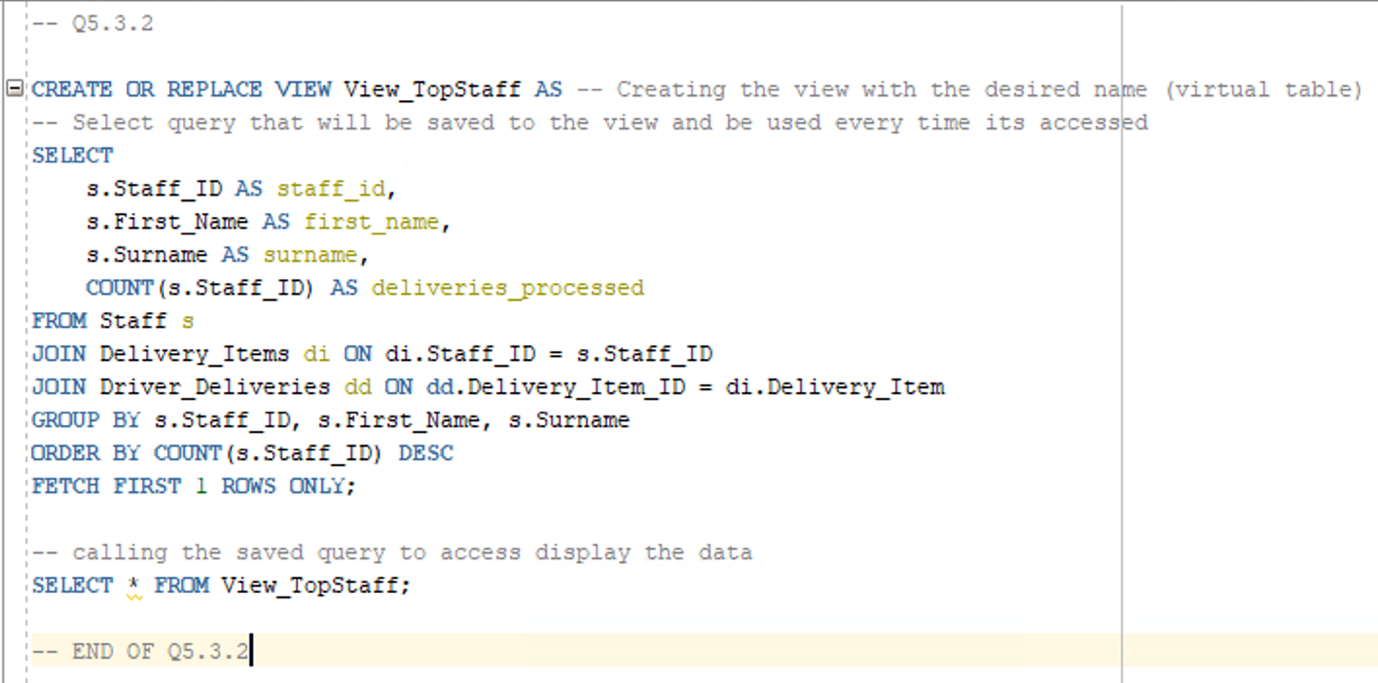
FETCH FIRST 1 ROWS ONLY;

-- calling the saved query to access display the data

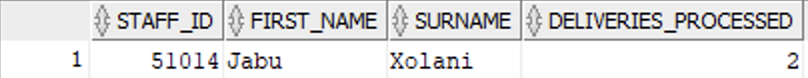
SELECT \* FROM View\_TopStaff;

-- END OF Q5.3.2

### Screenshot of SQL Code



### Screenshot of Successful Output



# Question 6

## Q6.1: Implicit and Explicit Cursors

### Motivations for Implicit Cursors

Implicit cursors are database tools that help simplify code by running operations automatically in the background (GeeksforGeeks, 2025). GeeksforGeeks (2025) mentions that “when we run a command to fetch information, the implicit cursor automatically processes the data one step at a time, starting from the beginning and going to the end.” Essentially, all work is done for developers by the system.

This allows for less code, automatic handling, less maintenance, and simple efficient queries (GeeksforGeeks, 2025); this makes using implicit cursors very beneficial as it saves lots of time and is highly accurate and efficient.

In context to Cheetah Deliveries, this is highly beneficial as the business uses vehicles for all their deliveries, meaning that the mileage is constantly being increased. Hence, having an implicit cursor for this use would mean that it ready to use and be executed after each delivery.

### Motivation for Explicit Cursors

Explicit cursors are database tools that developers control directly (GeeksforGeeks, 2025). “With explicit cursors, we can specify what data to look at and how to handle it” (GeeksforGeeks, 2025). GeeksforGeeks (2025) further mentions that these cursors take more effort to setup, however, they provide excellent management and operations of data.

These cursors provide greater control, flexibility, error handling, and multiple row handling (GeeksforGeeks, 2025); this makes using explicit cursors very beneficial as it handles complex data very effectively and does so to our own extent.

With regards to Cheetah Deliveries, this is very beneficial as there are multiple deliveries being processed on a daily basis and using an explicit cursor can help with operations regarding which staff member processed each delivery.

### Implicit Cursor Code

-- Implicit Cursor

SET SERVEROUTPUT ON;

DECLARE var\_rows NUMBER(5);

BEGIN

UPDATE Vehicle

SET Mileage = Mileage + 100;

IF SQL%NOTFOUND THEN

DBMS\_OUTPUT.PUT\_LINE('No vehicle mileage updated');

ELSE var\_rows := SQL%ROWCOUNT;

DBMS\_OUTPUT.PUT\_LINE('Mileage for ' || var\_rows || ' vehicles updated');

END IF;

END;

/

### Explicit Cursor Code

-- Explicit Cursor

-- This cursor will show each staff member that processed an order and how many total orders have been processed at each iteration

SET SERVEROUTPUT ON;

DECLARE

CURSOR curs\_staff IS

SELECT s.Staff\_ID, s.First\_Name, s.Surname

FROM Staff s

JOIN Delivery\_Items di ON di.Staff\_ID = s.Staff\_ID;

v\_staff curs\_staff%ROWTYPE;

BEGIN

OPEN curs\_staff;

DBMS\_OUTPUT.PUT\_LINE('--- Staff Deliveries Processed ---');

LOOP

FETCH curs\_staff INTO v\_staff;

EXIT WHEN curs\_staff%NOTFOUND;

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

DBMS\_OUTPUT.PUT\_LINE('Staff: ' || v\_staff.First\_Name || ' ' || v\_staff.Surname);

DBMS\_OUTPUT.PUT\_LINE('Rows fetched so far: ' || curs\_staff%ROWCOUNT);

END LOOP;

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

CLOSE curs\_staff;

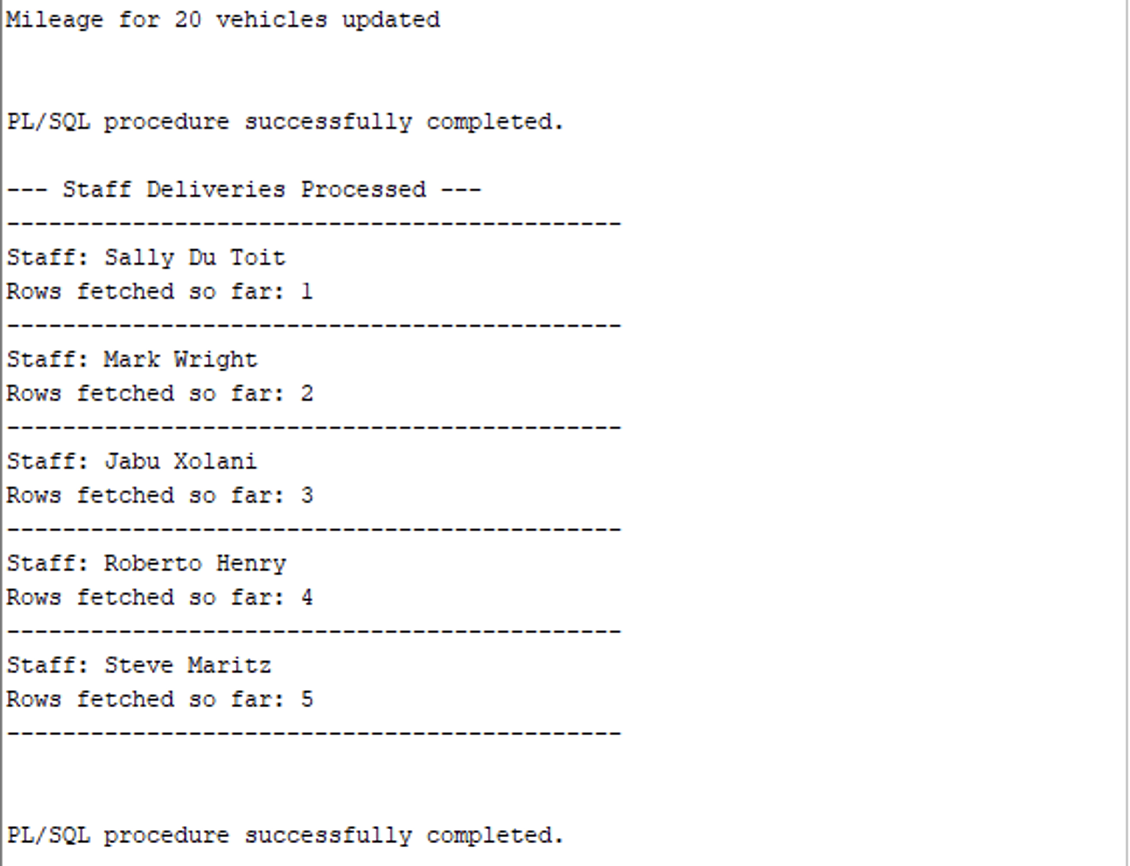
END;

/

### Screenshot of Code in SQL Developer



### Screenshot of Successful Cursor Outputs



## Q6.2: Using a Sequence

Sequences are “used to generate unique numeric values in a sequential order” (GeeksforGeeks, 2025). GeeksforGeeks (2025) further mentions that these are widely used to generate primary keys. This aids developers as they won’t need to manually insert the next unique value as a primary key for a new instance. Therefore, Cheetah Deliveries could use a sequence for each table to provide primary key values to new instances of an entity, rather than having to manually write out the primary key.

The example will be for new entries into Billing:

### SQL Code

-- Q6.2

CREATE SEQUENCE seq\_bill START WITH 821 INCREMENT BY 1;

INSERT INTO Billing (Bill\_ID, Customer\_ID, Staff\_ID, Bill\_Date)

VALUES (seq\_bill.NEXTVAL, 11011, 51013, TO\_DATE('09-Dec-22', 'DD-Mon-YY'));

-- querying new insert

SELECT \* FROM Billing WHERE Bill\_ID=821;

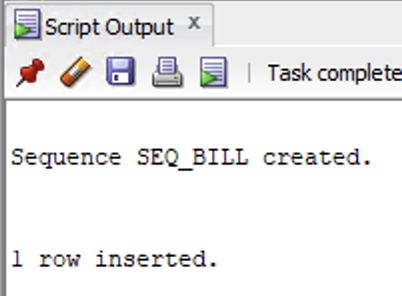
-- END OF Q6.2

### Screenshot of Code

A close-up of a computer code

AI-generated content may be incorrect.

### Screenshot of Successful Implementation



A close up of a number

AI-generated content may be incorrect.

# References

DatabaseTown. 2025. Relational Database vs Flat File (Differences and Similiraties), 24 January 2023. [Online]. Available at: <https://databasetown.com/relational-database-vs-flat-file-differences-similarities/> [Accessed 24 September 2025].

GeeksforGeeks. 2025. Difference between Implicit and Explicit Cursors, 12 July 2025. [Online]. Available at: <https://www.geeksforgeeks.org/dbms/difference-between-implicit-and-explicit-cursors/> [Accessed 25 September 2025].

GeeksforGeeks. 2025. SQL | Sequences, 13 January 2025. [Online]. Available at: <https://www.geeksforgeeks.org/sql/sql-sequences/> [Accessed 25 September 2025].

GeeksforGeeks. 2025. SQL Views, 8 September 2025. [Online]. Available at: <https://www.geeksforgeeks.org/sql/sql-views/> [Accessed 24 September 2025].

Hanna, K, T. 2025. What is segregation of duties (SoD)?, 6 May 2025. Available at: <https://www.techtarget.com/whatis/definition/segregation-of-duties-SoD> [Accessed 23 September 2025].

Oracle. 2025. How to Create Users, Grant Them Privileges, and Remove Them in Oracle Database, 30 July 2018. [Online]. Available at: <https://blogs.oracle.com/sql/post/how-to-create-users-grant-them-privileges-and-remove-them-in-oracle-database> [Accessed 23 September 2025].

Rosenzweig, B. Rakhimov, E. 2025. PL/SQL Concepts, 3 July 2023. [Online]. Available at: <https://www.pearsonitcertification.com/articles/article.aspx?p=3178919> [Accessed 24 September 2025].