GROUP 16

SEG 1201 FINAL PROJECT PRESENTATION

Reporting on our progress and project as a whole.

Todays Agenda

Highlights and key areas of discussion

01

Introduction

03

Implementation

02

Database Design

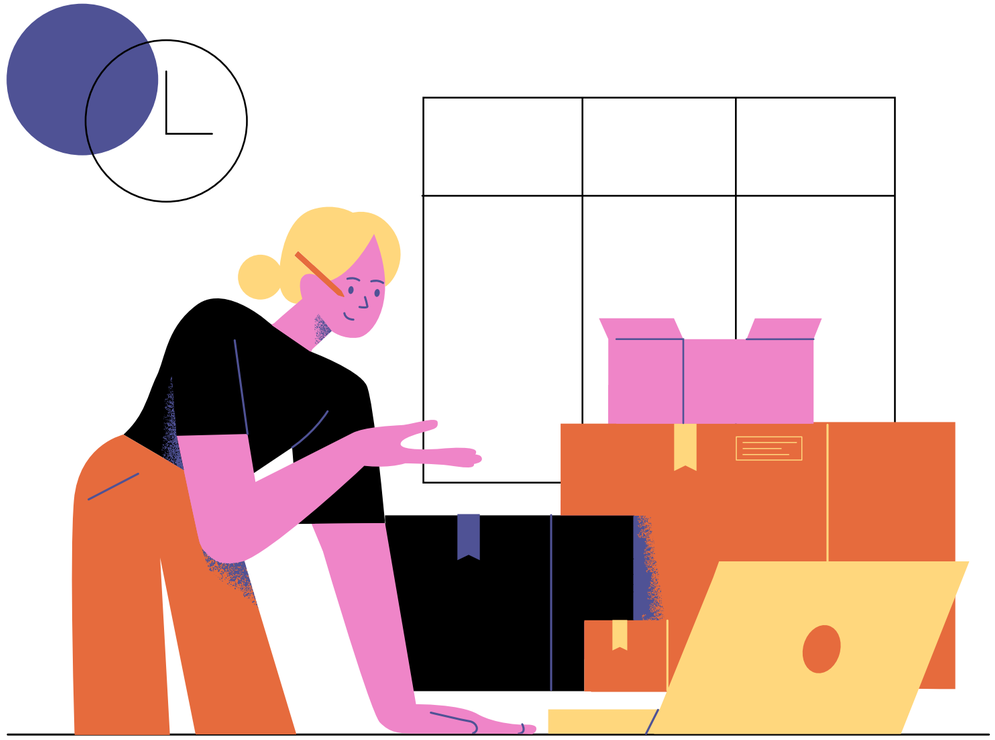
04

Queries

PROJECT PLAN

TASKS	WEEK 1	WEEK 2	WEEK 3	WEEK 4	WEEK 5
CONSTRUCTING CASE SCENARIO	✓				
CREATING BUSINESS RULES	✓				
CREATING FLOWCHART, ERD AND RDM		✓			
SCRIPT		✓			
VIEWS AND INDEX			✓		
· USER QUERIES			✓		
·SQL STATEMENTS AND RESULTS				✓	
PRESENTATION				✓	✓

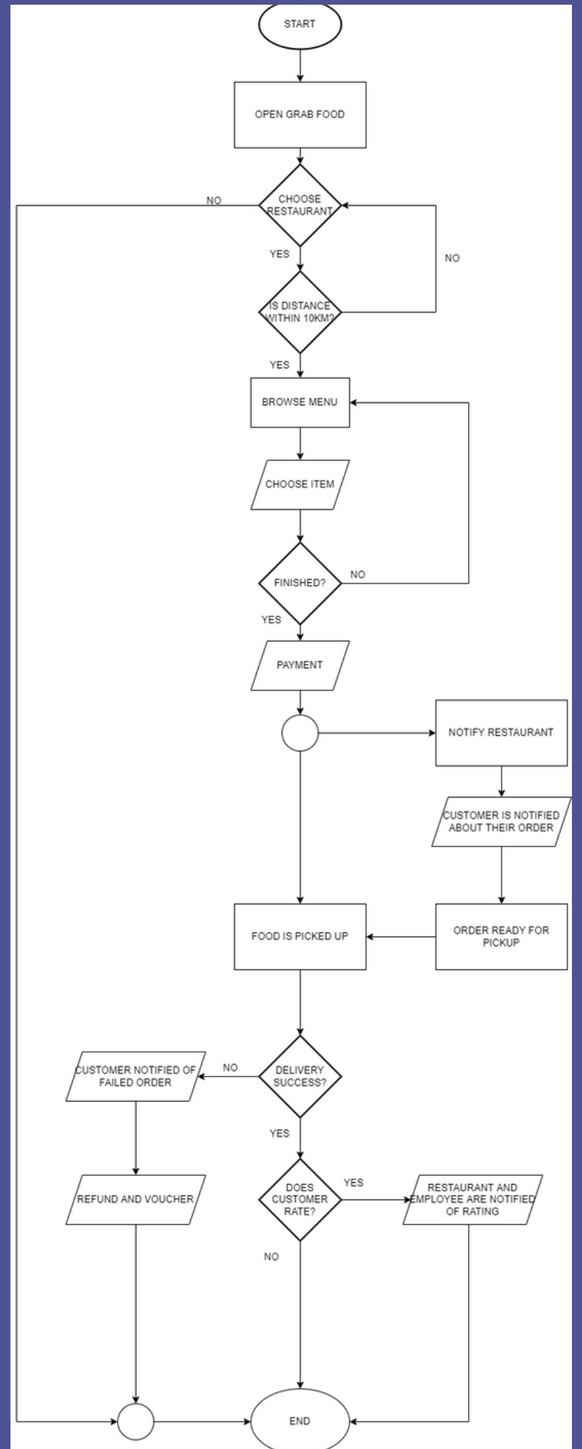
Our Scenario



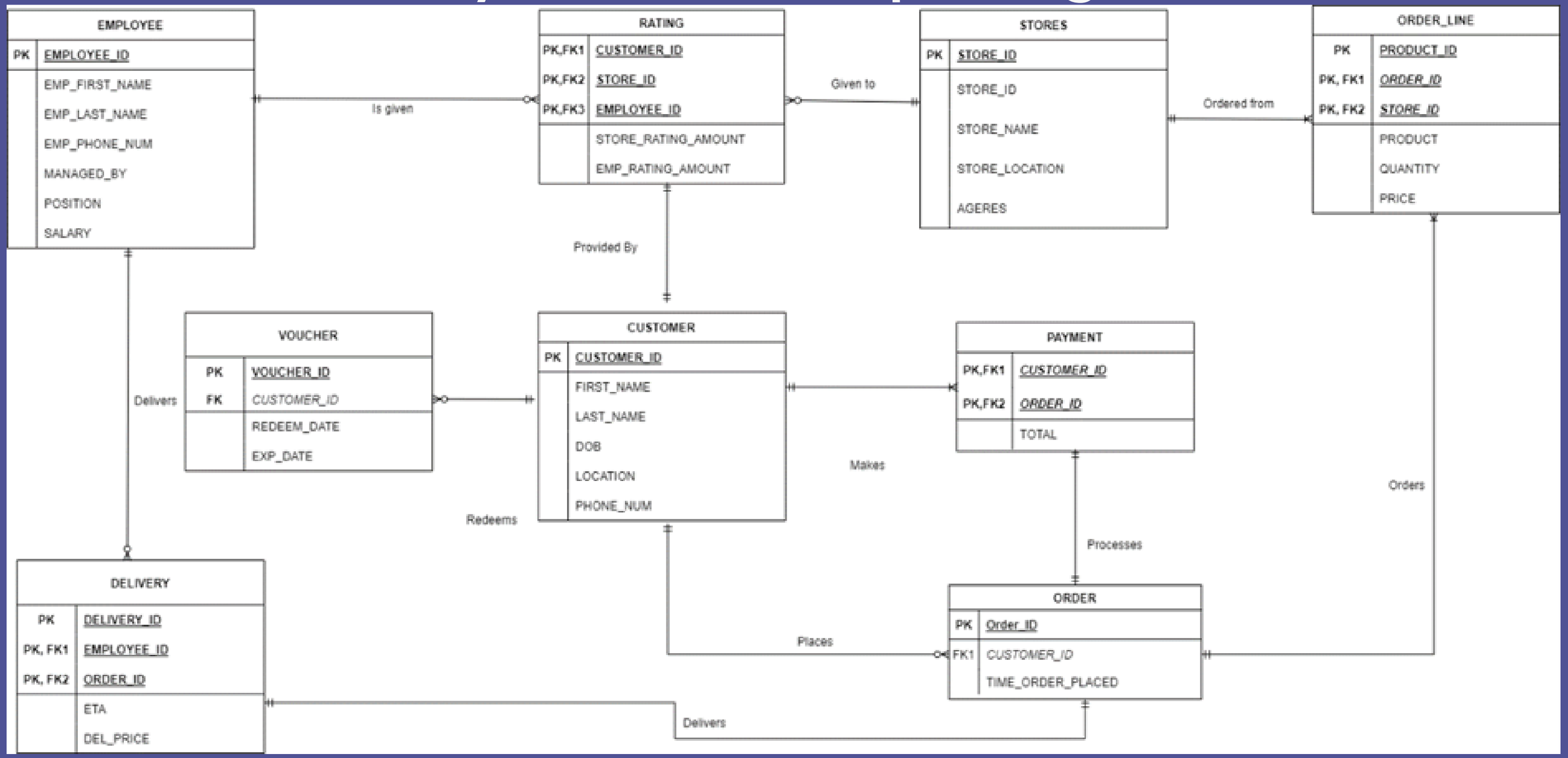
Due to the pandemic in 2020,

There was an increase of complaints from the customers. This is because restaurants were taking in too many orders at once, furthermore, delivery drivers were handling too many orders at once. To prevent this problem from exacerbating further, Grab has hired new database designers to control the influx of consumer data while ensuring consumer data remains consistent and up to date

Flowchart of the User Experience



Entity Relationship Diagram



GrabFood

Demonstration of the Script

We will show our script now

GrabFood

Views and Indexes

View (1)

Create View For Customer Age

```
CREATE VIEW customer_age AS
SELECT CUSTOMER_ID AS ID, FIRST_NAME, LAST_NAME, ((EXTRACT (YEAR
FROM CURRENT_DATE)) - (EXTRACT (YEAR FROM DOB))) AS Age
FROM Customer
```

```
1  SELECT *
2  FROM customer_age
```

Results | Explain | Describe | Saved SQL | History

ID	FIRST_NAME	LAST_NAME	AGE
111	Ashley	Lee	23
123	Jason	Yu	20
189	Muhammad	Arif	22
222	John	Doe	19
235	Ashley	Lee	32
333	Muthu	Segaran	28
432	Ashley	McKinsey	22
444	Alex	Tan	32
456	Aly	Kew	23
457	Justin	Yu	66
478	Muhammad	Aqeel	32
516	Ali	Ahsan	12
666	Jody	Yu	51
777	Daniel	Amin	32
790	Muhammad	Areef	22
883	Muhammad	Zahar	33
886	Laura	McKinsey	22
888	Marcus	Tan	14
1001	Nur	Aisha	14
1002	Nur	Salma	16

View (2)

Create View For Employee Average Rating_

```
CREATE VIEW employee_average_ratings AS
SELECT Employee.EMPLOYEE_ID AS Employee,
ROUND((AVG(RATING.EMP_RATING_AMOUNT)),1) AS Rating_Amount
FROM Employee, Rating
WHERE EMPLOYEE.EMPLOYEE_ID = RATING.EMPLOYEE_ID
GROUP BY EMPLOYEE.EMPLOYEE_ID
```

```
1 SELECT *
2 FROM employee_average_ratings
```

Results | Explain | Describe | Saved SQL | History

EMPLOYEE	RATING_AMOUNT
A48	4.5
A10	4.5
A17	3.3
A49	5
A01	3.7
A02	3
A12	3.5

7 rows returned in 0.02 seconds Download

View (3)

Create View For Store Average Rating_

```
CREATE VIEW store_average_ratings AS
SELECT STORES.STORE_ID AS ID, ROUND((AVG (RATING.STORE_RATING_AMOUNT)),1) AS
Rating_Amount
FROM Stores, Rating
WHERE Stores.Store_ID = Rating.Store_ID
GROUP BY Stores.Store_ID
```

```
1  SELECT *
2  FROM store_average_ratings
```

Results | Explain | Describe | Saved SQL | History

ID	RATING_AMOUNT
6	3
14	2
1	3.3
11	2
12	4
17	3.5
3	2
16	4

Index

```
CREATE INDEX CUSTOMER_AGE on CUSTOMER (CUSTOMER_ID, FIRST_NAME,
LAST_NAME, DOB);

CREATE INDEX STORE_RATING on STORES (STORE_ID, STORE_NAME, AGERES);

CREATE INDEX EMPLOYEE_RATING on EMPLOYEE (EMPLOYEE_ID,
EMP_FIRST_NAME, EMP_LAST_NAME);
```

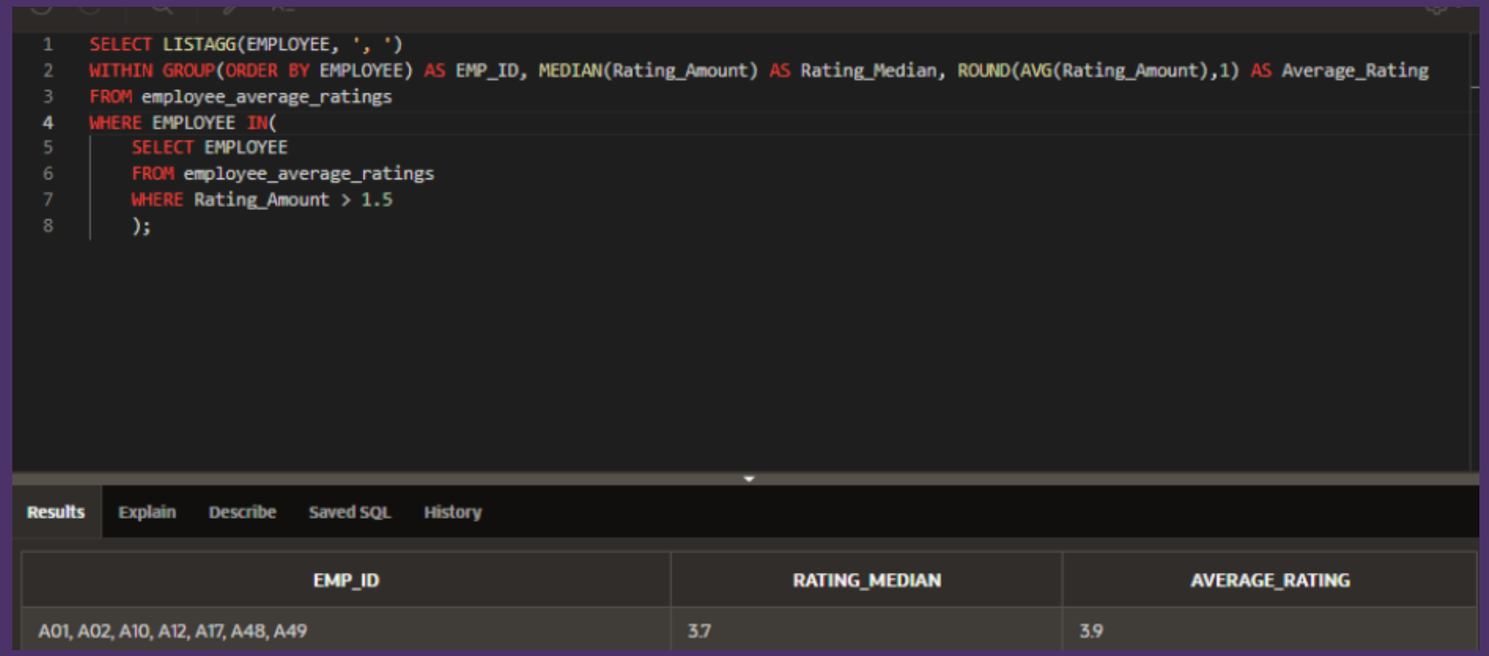
GrabFood

User Queries

1.Grab wishes to analyse employee's rating during the pandemic for their year-end report, so the statistics team from grab food wishes to see the delivery
employees and the discrepancy between the median rating, and average rating.

CODE:

```
SELECT LISTAGG(EMPLOYEE, ', ')
WITHIN GROUP(ORDER BY EMPLOYEE) AS EMP_ID,
MEDIAN(Rating_Amount) AS Rating_Median,
ROUND(AVG(Rating_Amount),1) AS Average_Rating
FROM employee_average_ratings
WHERE EMPLOYEE IN(
 SELECT EMPLOYEE
 FROM employee_average_ratings
 WHERE Rating_Amount > 1.5
 );
```



EMP_ID	RATING_MEDIAN	AVERAGE_RATING
A01, A02, A10, A12, A17, A48, A49	3.7	3.9

2.Grab wishes to find the most active employee in the Selangor area to give them a reward. As such, GrabFood has given the database designers a task to find a list of the most active employees in terms of delivery in Selangor.

CODE:

```
SELECT Employee.EMPLOYEE_ID,
Employee.EMP_FIRST_NAME,Employee.EMP_LAST_NAME,
Customer.CLOCATION,
COUNT(DELIVERY.EMPLOYEE_ID) AS
NUMBER_OF_COUNT
FROM Employee
JOIN Delivery ON Employee.EMPLOYEE_ID =
Delivery.EMPLOYEE_ID
JOIN Orders ON Delivery.ORDER_ID = Orders.ORDER_ID
JOIN Customer ON Orders.Customer_ID =
Customer.CUSTOMER_ID
WHERE Customer.CLOCATION LIKE '%Selangor%'
GROUP BY Employee.EMPLOYEE_ID,
Employee.EMP_FIRST_NAME,Employee.EMP_LAST_NAME,
Customer.CLOCATION
HAVING COUNT(Orders.ORDER_ID) > 2
```

```
SELECT Employee.EMPLOYEE_ID, Employee.EMP_FIRST_NAME,Employee.EMP_LAST_NAME, Customer.CLOCATION, COUNT(DELIVERY.EMPLOYEE_ID) AS NUMBER_OF_COUNT
FROM Employee
JOIN Delivery ON Employee.EMPLOYEE_ID = Delivery.EMPLOYEE_ID
JOIN Orders ON Delivery.ORDER_ID = Orders.ORDER_ID
JOIN Customer ON Orders.Customer_ID = Customer.CUSTOMER_ID
WHERE Customer.CLOCATION LIKE '%Selangor%'
GROUP BY Employee.EMPLOYEE_ID, Employee.EMP_FIRST_NAME,Employee.EMP_LAST_NAME, Customer.CLOCATION
HAVING COUNT(Orders.ORDER_ID) > 2
```

Results | Explain | Describe | Saved SQL | History

EMPLOYEE_ID	EMP_FIRST_NAME	EMP_LAST_NAME	CLOCATION	NUMBER_OF_COUNT
A01	Ali	Darish	34, Persiaran Subang, Taman Indah Subang Uep, 47619 Subang Jaya, Selangor, Malaysia	3

3.Grab wishes to find employees that have not done a delivery, or any employees that have recently (In the past 3 months) not done any deliveries in order to take disciplinary actions towards them.

CODE:

```
SELECT Delivery.DELIVERY_ID, Employee.EMP_LAST_NAME,
Employee.EMP_FIRST_NAME
FROM Delivery
RIGHT JOIN Employee
ON Delivery.EMPLOYEE_ID = Employee.EMPLOYEE_ID
WHERE
Employee.POSITION LIKE 'Delivery' AND DELIVERY.DELIVERY_ID
IS NULL OR DELIVERY.ETA < '01-APR-2021'
ORDER BY Delivery.DELIVERY_ID;
```

```
SELECT Delivery.DELIVERY_ID, Employee.EMP_LAST_NAME, Employee.EMP_FIRST_NAME
FROM Delivery
RIGHT JOIN Employee
ON Delivery.EMPLOYEE_ID = Employee.EMPLOYEE_ID
WHERE Employee.POSITION LIKE 'Delivery' AND DELIVERY.DELIVERY_ID IS NULL OR DELIVERY.ETA < '01-APR-2021' ORDER BY Delivery.DELIVERY_ID;
```

Results | Explain | Describe | Saved SQL | History

DELIVERY_ID	EMP_LAST_NAME	EMP_FIRST_NAME
27	Ashraf	Aliya
32	Darish	Ellie
35	Raju	Muthu
38	Kim	Ahmad
-	Tan	Peter
-	Kim	Alia
-	Wong	Alya
-	Kim	Larry

4.Grab wishes to find the managers of each employee, to easily find out which manager to contact in the situation where disciplinary action needs to be taken.

CODE:

```
SELECT MANAGER.EMP_FIRST_NAME AS MANAGER,
LISTAGG(EMP.EMPLOYEE_ID, ', ') WITHIN GROUP(ORDER BY
EMP.EMPLOYEE_ID) AS Employees
FROM EMPLOYEE EMP, EMPLOYEE MANAGER
WHERE EMP.MANAGED_BY = MANAGER.EMPLOYEE_ID
GROUP BY MANAGER.EMP_FIRST_NAME
```

```
1  SELECT MANAGER.EMP_FIRST_NAME AS MANAGER, LISTAGG(EMP.EMPLOYEE_ID, ', ') WITHIN GROUP(ORDER BY EMP.EMPLOYEE_ID) AS Employees
2  FROM EMPLOYEE EMP, EMPLOYEE MANAGER
3  WHERE EMP.MANAGED_BY = MANAGER.EMPLOYEE_ID
4  GROUP BY MANAGER.EMP_FIRST_NAME
5
```

esults | Explain | Describe | Saved SQL | History

MANAGER	EMPLOYEES
Aisha	A02, A08, A12, A19, A26, A34, A39, A45, A50
Emily	A04, A09, A15, A22, A29, A35, A42, A46
Justin	A05, A11, A16, A23, A31, A38, A43, A47

Thank you for listening!