

Phase 8

Toronto Metropolitan University

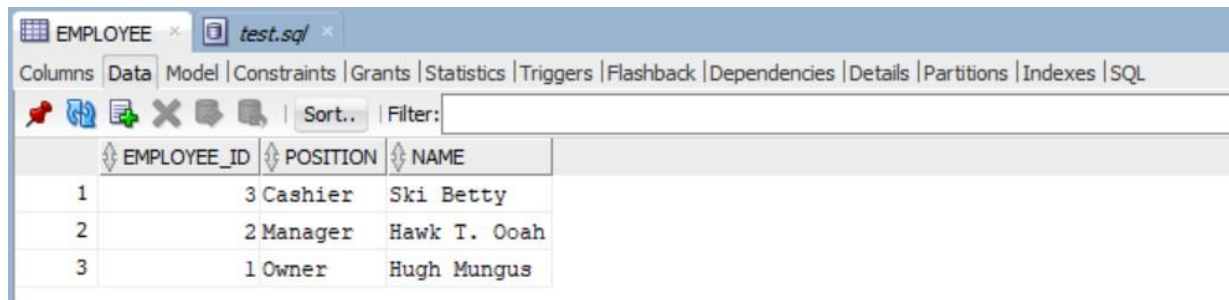
Simon Lin (501103322),  
Dylan Ha (501056670),  
Enes Polat (501061594)

CPS510 - Database Systems

Point of Sale System for Shopper Drug Marts

## Database Normalization/BCNF:

Employee (Employee\_ID#, Position, Name)



The screenshot shows a database management interface with a tab for 'EMPLOYEE' and a file named 'test.sql'. The 'Columns' tab is active, displaying the table structure with three columns: EMPLOYEE\_ID, POSITION, and NAME. Below the column headers, three rows of data are listed.

	EMPLOYEE_ID	POSITION	NAME
1	3	Cashier	Ski Betty
2	2	Manager	Hawk T. Ooah
3	1	Owner	Hugh Mungus

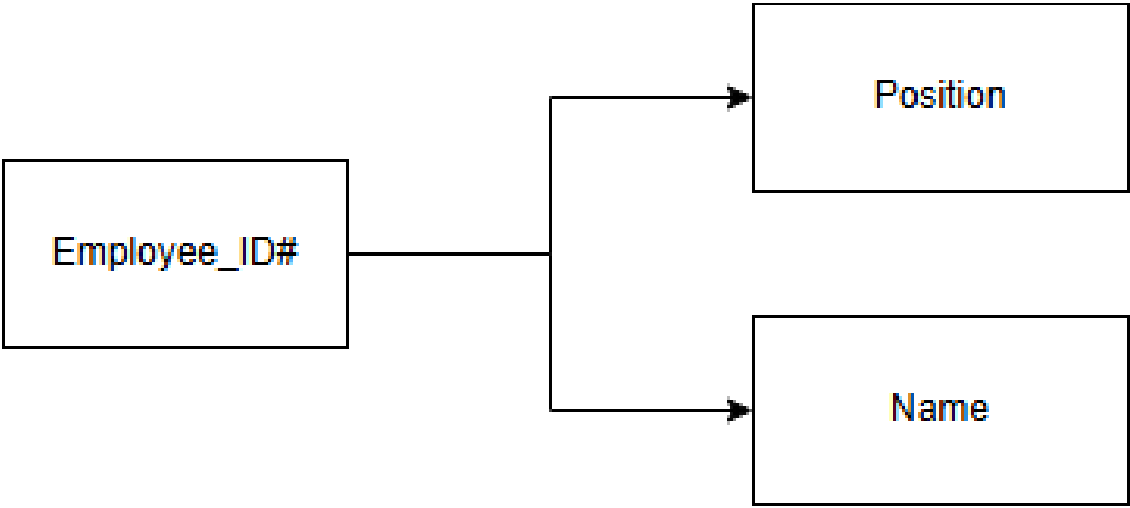
Functional Dependencies:

$\{\text{Employee\_ID}\} \rightarrow \text{Position}$

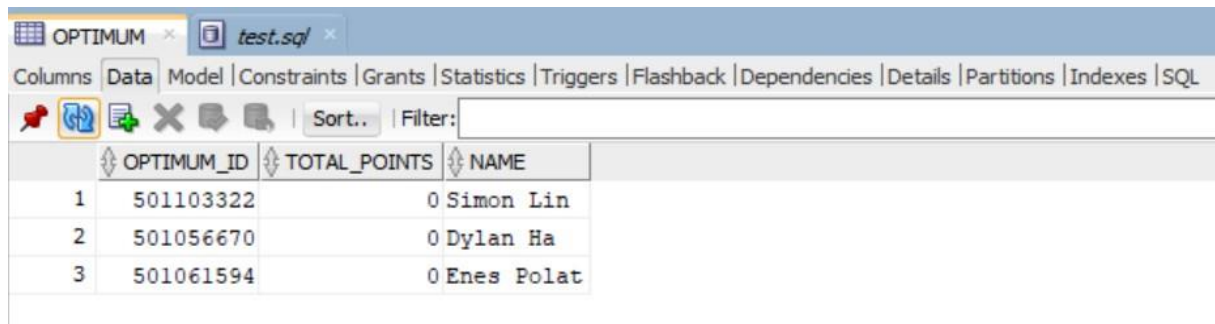
$\{\text{Employee\_ID}\} \rightarrow \text{Name}$

Bernstein's Algorithm:

1. Minimize the list of FDs ( $\text{FD } X \rightarrow Y, Z$ )
  - a. FD1:  $\{\text{Employee\_ID}\} \rightarrow \text{Position}$
  - b. FD2:  $\{\text{Employee\_ID}\} \rightarrow \text{Name}$
  - c. Combine the FDs to form a single FD
    - i.  $\{\text{Employee\_ID}\} \rightarrow \text{Position, Name}$
2. Get rid of redundant FDs
  - a. Since we combined FD1 and FD2 into a single dependency, there are no redundant dependencies to remove here.
3. Minimize left hand side
  - a. The left hand side is just Employee\_ID, which is already the sole candidate key and superkey in the relation. No further decomposition is needed.
4. Derive final schema
  - a. We derive the final schema of  
Employee (Employee\_ID#, Position, Name)  
Employee\_ID is the candidate key and determines all other attributes in the relation, therefore the relation satisfies BCNF and 3NF.



Optimum(Optimum\_ID#, Total\_Points, Name)



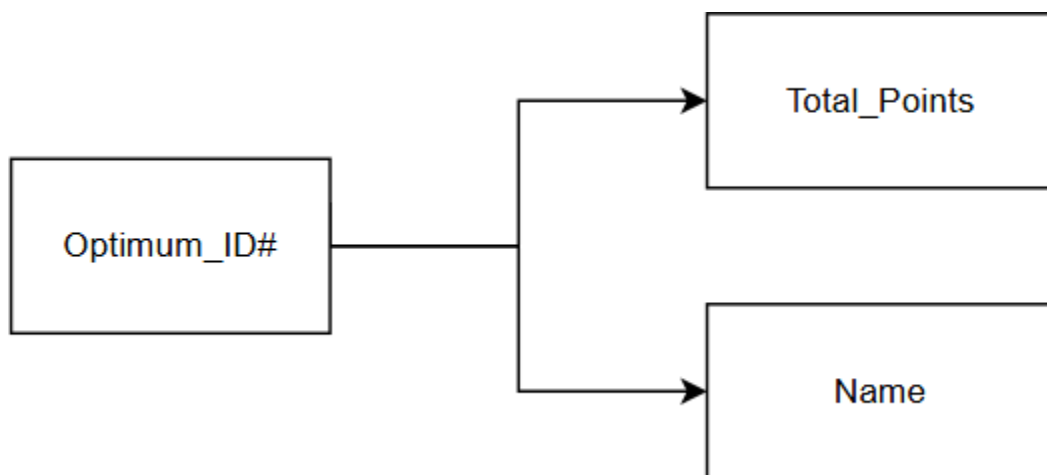
	OPTIMUM_ID	TOTAL_POINTS	NAME
1	501103322	0	Simon Lin
2	501056670	0	Dylan Ha
3	501061594	0	Enes Polat

Functional Dependencies:

$\{\text{Optimum\_ID}\} \rightarrow \text{Total\_Points}$   
 $\{\text{Optimum\_ID}\} \rightarrow \text{Name}$

Bernstein's Algorithm:

1. Minimize the list of FDs ( $\text{FD } X \rightarrow Y, Z$ )
  - b. FD1:  $\{\text{Optimum\_ID}\} \rightarrow \text{Total\_Points}$
  - c. FD2:  $\{\text{Optimum\_ID}\} \rightarrow \text{Name}$
  - d. Combine the FDs to form a single FD
    - i.  $\{\text{Optimum\_ID}\} \rightarrow \text{Total\_Points, Name}$
2. Get rid of redundant FDs
  - e. Since we combined FD1 and FD2 into a single dependency, there are no redundant dependencies to remove here.
3. Minimize left hand side
  - f. The left hand side is just Optimum\_ID, which is already the sole candidate key and superkey in the relation. No further decomposition is needed.
4. Derive final schema
  - g. We derive the final schema of  
Optimum(Optimum\_ID#, Total\_Points, Name)  
Optimum\_ID is the candidate key and determines all other attributes in the relation, therefore the relation satisfies BCNF and 3NF.



Customer(Customer\_ID#, Optimum\_ID, Name)

CUSTOMER			
Columns Data Model Constraints Grants Statistics Triggers Flashback Dependencies Details Partitions Indexes SQL			
	CUSTOMER_ID	OPTIMUM_ID	NAME
1	1	501103322	Simon Lin
2	2	501056670	Dylan Ha
3	3	501061594	Enes Polat

Functional Dependencies:

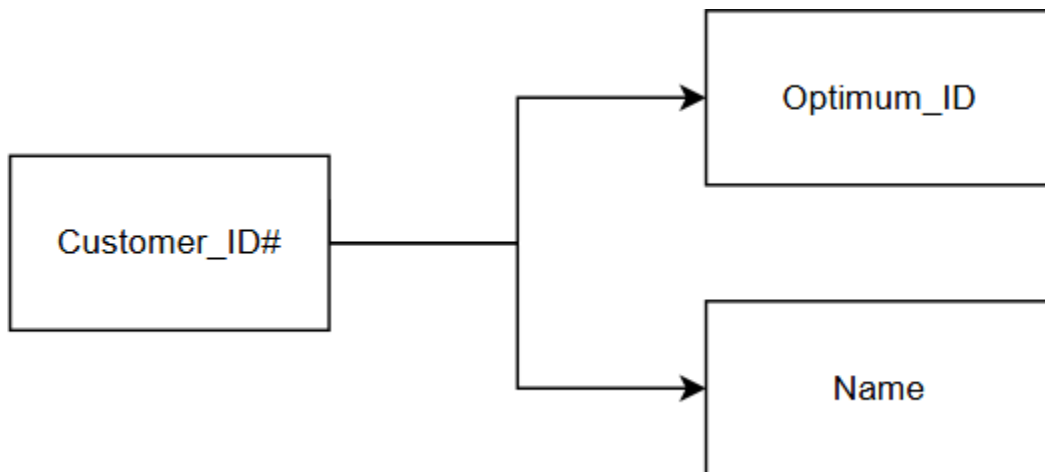
$\{ \text{Customer\_ID} \} \rightarrow \text{Optimum\_ID}$

$\{ \text{Customer\_ID} \} \rightarrow \text{Name}$

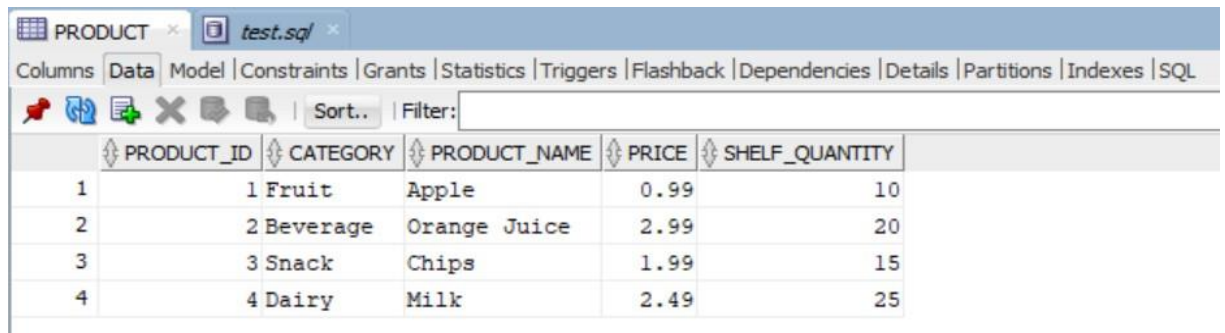
Bernstein's Algorithm:

1. Minimize the list of FDs (FD  $X \rightarrow Y, Z$ )

- a. FD1:  $\{\text{Customer\_ID}\} \rightarrow \text{Optimum\_ID}$
  - b. FD2:  $\{\text{Customer\_ID}\} \rightarrow \text{Name}$
  - c. Combine the FDs to form a single FD
    - i.  $\{\text{Customer\_ID}\} \rightarrow \text{Optimum\_ID, Name}$
2. Get rid of redundant FDs
- a. Since we combined FD1 and FD2 into a single dependency, there are no redundant dependencies to remove here.
3. Minimize left hand side
- a. The left hand side is just Customer\_ID, which is already the sole candidate key and superkey in the relation. No further decomposition is needed.
4. Derive final schema
- a. We derive the final schema of  
Customer(Customer\_ID#, Optimum\_ID, Name)  
Customer\_ID is the candidate key and determines all other attributes in the relation, therefore the relation satisfies BCNF and 3NF.



Product(Product\_ID#, Category, Product\_Name, Price, Shelf\_Quantity)



The screenshot shows a database management interface with a tab labeled 'PRODUCT' and a file named 'test.sql'. Below the tabs is a toolbar with icons for various database operations. The main area displays a table with the following columns: PRODUCT\_ID, CATEGORY, PRODUCT\_NAME, PRICE, and SHELF\_QUANTITY. The table contains four rows of data.

	PRODUCT_ID	CATEGORY	PRODUCT_NAME	PRICE	SHELF_QUANTITY
1	1	Fruit	Apple	0.99	10
2	2	Beverage	Orange Juice	2.99	20
3	3	Snack	Chips	1.99	15
4	4	Dairy	Milk	2.49	25

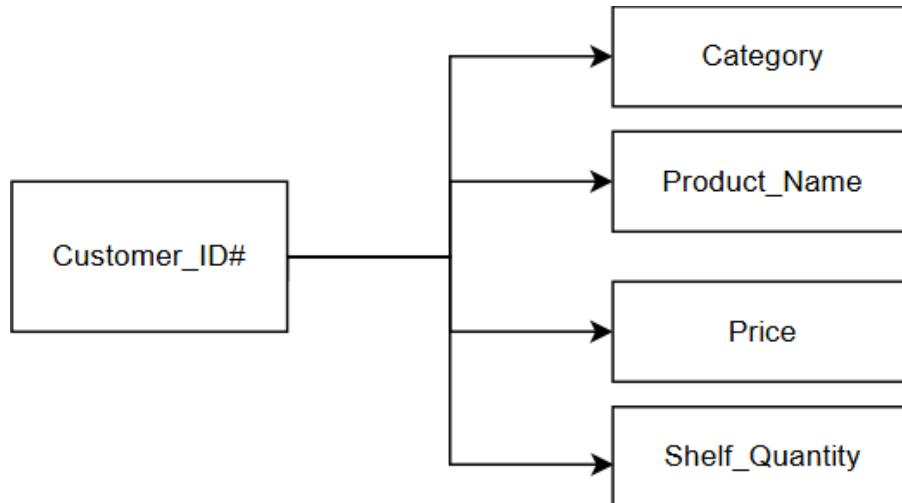
Functional Dependencies:

$\{\text{Product\_ID}\} \rightarrow \text{Category}$   
 $\{\text{Product\_ID}\} \rightarrow \text{Product\_Name}$   
 $\{\text{Product\_ID}\} \rightarrow \text{Price}$   
 $\{\text{Product\_ID}\} \rightarrow \text{Shelf\_Quantity}$

Bernstein's Algorithm:

1. Minimize the list of FDs ( $FD X \rightarrow Y, Z$ )
  - a. FD1:  $\{Product\_ID\} \rightarrow Category$
  - b. FD2:  $\{Product\_ID\} \rightarrow Product\_Name$
  - c. FD3:  $\{Product\_ID\} \rightarrow Price$
  - d. FD4:  $\{Product\_ID\} \rightarrow Shelf\_Quantity$
  - e. Combine the FDs to form a single FD
    - ii.  $\{Product\_ID\} \rightarrow Category, Product\_Name, Price, Shelf\_Quantity$
2. Get rid of redundant FDs
  - a. Since we combined FD1, FD2, FD3 and FD4 into a single dependency, there are no redundant dependencies to remove here.
3. Minimize left hand side
  - a. The left hand side is just Product\_ID, which is already the sole candidate key and superkey in the relation. No further decomposition is needed.
4. Derive final schema
  - a. We derive the final schema of  $Product(\underline{Product\_ID\#}, Category, Product\_Name, Price, Shelf\_Quantity)$   
Product\_ID is the candidate key and determines all other attributes in the relation, therefore the relation satisfies BCNF and 3NF.





Transaction(Transaction\_ID#, Employee\_ID#, Total\_PointsSpent, Total\_Price, Payment\_Method, Transaction\_Date)

TRANSACTION						
Columns   Data   Model   Constraints   Grants   Statistics   Triggers   Flashback   Dependencies   Details   Partitions   Indexes						
	TRANSACTION_ID	EMPLOYEE...	TOTAL_P...	TOTAL_P...	PAYMENT...	TRANSACTION_DATE
1	1	3	500	23.59	Cash	24-SEP-24

Functional Dependencies:

$\{Transaction\_ID\} \rightarrow Employee\_ID$   
 $\{Transaction\_ID\} \rightarrow Total\_PointsSpent$   
 $\{Transaction\_ID\} \rightarrow Total\_Price$   
 $\{Transaction\_ID\} \rightarrow Payment\_Method$   
 $\{Transaction\_ID\} \rightarrow Transaction\_Date$

Bernstein's Algorithm:

1. Minimize the list of FDs (FD  $X \rightarrow Y, Z$ )
  - a. FD1:  $\{Transaction\_ID\} \rightarrow Employee\_ID$
  - b. FD2:  $\{Transaction\_ID\} \rightarrow Total\_PointsSpent$

- c. FD3: {Transaction\_ID} → Total\_Price
- d. FD4: {Transaction\_ID} → Payment\_Method
- e. FD5: {Transaction\_ID} → Transaction\_Date
- f. Combine the FDs to form a single FD
  - iii. {Transaction\_ID} → Employee\_ID, Total\_PointsSpent, Total\_Price, Payment\_Method, Transaction\_Date

2. Get rid of redundant FDs

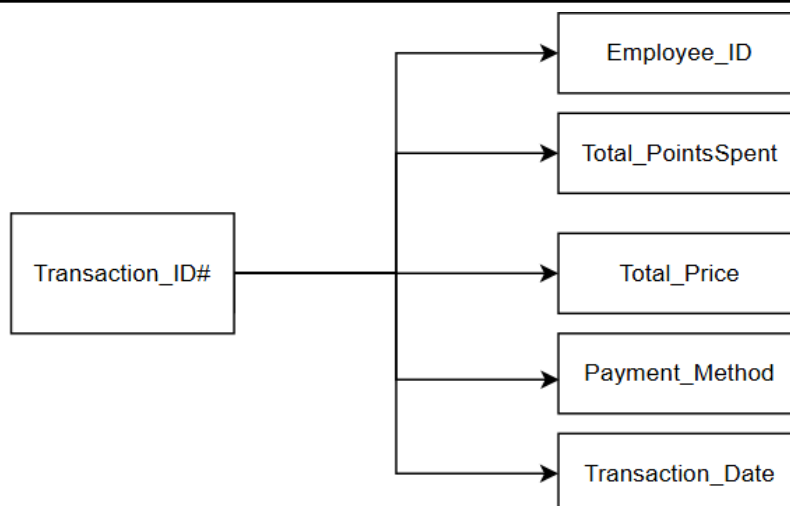
- b. Since we combined FD1, FD2, FD3, FD4 and FD5 into a single dependency, there are no redundant dependencies to remove here.

3. Minimize left hand side

- b. The left hand side is just Transaction\_ID, which is already the sole candidate key and superkey in the relation. No further decomposition is needed.

4. Derive final schema

- b. We derive the final schema of  
Transaction(Transaction\_ID#, Employee\_ID#, Total\_PointsSpent, Total\_Price, Payment\_Method, Transaction\_Date)  
Transaction\_ID is the candidate key and determines all other attributes in the relation, therefore the relation satisfies BCNF and 3NF.



Receipt(Transaction\_ID#, Product\_List, Points\_Earned, Total\_Price, Payment\_Method, Transaction\_Date)

TRANSAC...	PRODUCT_LIST	POINTS_E...	TOTAL_P...	PAYMENT...	TRANSACTION_DATE
1	1 x22 Apples	500	23.59	Cash	24-SEP-24

Functional Dependencies:

$\{ \text{Transaction\_ID} \} \rightarrow \text{Product\_List}$   
 $\{ \text{Transaction\_ID} \} \rightarrow \text{Points\_Earned}$   
 $\{ \text{Transaction\_ID} \} \rightarrow \text{Total\_Price}$   
 $\{ \text{Transaction\_ID} \} \rightarrow \text{Payment\_Method}$   
 $\{ \text{Transaction\_ID} \} \rightarrow \text{Transaction\_Date}$

Function Dependencies  
Removed Redundancy:

$\{ \text{Transaction\_ID} \} \rightarrow \text{Product\_List}$   
 $\{ \text{Transaction\_ID} \} \rightarrow \text{Points\_Earned}$

Bernstein's Algorithm:

1. Minimize the list of FDs ( $\text{FD } X \rightarrow Y, Z$ )
  - a. FD1:  $\{ \text{Transaction\_ID} \} \rightarrow \text{Product\_List}$
  - b. FD2:  $\{ \text{Transaction\_ID} \} \rightarrow \text{Points\_Earned}$
  - c. FD3:  $\{ \text{Transaction\_ID} \} \rightarrow \text{Total\_Price}$

- d. FD4: {Transaction\_ID} → Payment\_Method
- e. FD5: {Transaction\_ID} → Transaction\_Date
- g. Combine the FDs to form a single FD
  - iv. {Transaction\_ID} → Product\_List, Points\_Earned, Total\_Price, Payment\_Method, Transaction\_Date

2. Get rid of redundant FDs

Find X+ in the reduced list, If X+ contains Y, then X, Y is redundant

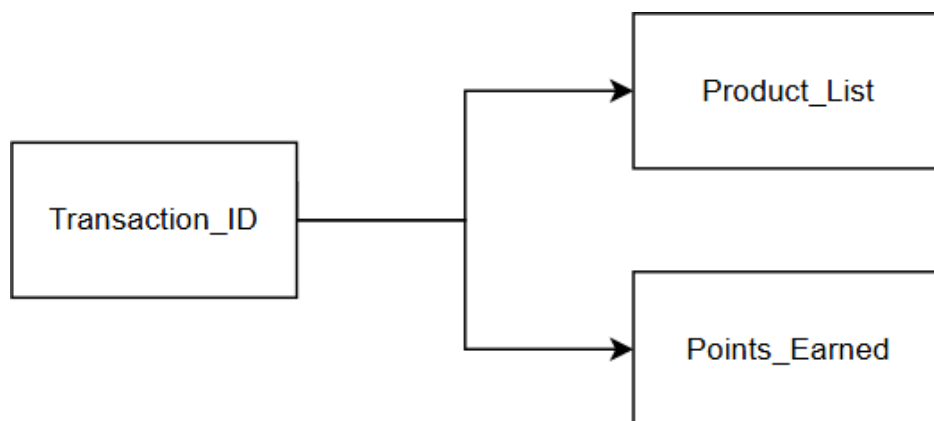
- a. We see that Total\_Price, Payment\_Method and Transaction\_Date are repeated FDs from the transaction table, we eliminate these dependencies from the table.
- b. This leaves {Transaction\_ID} → Product\_List, Points\_Earned

3. Minimize left hand side

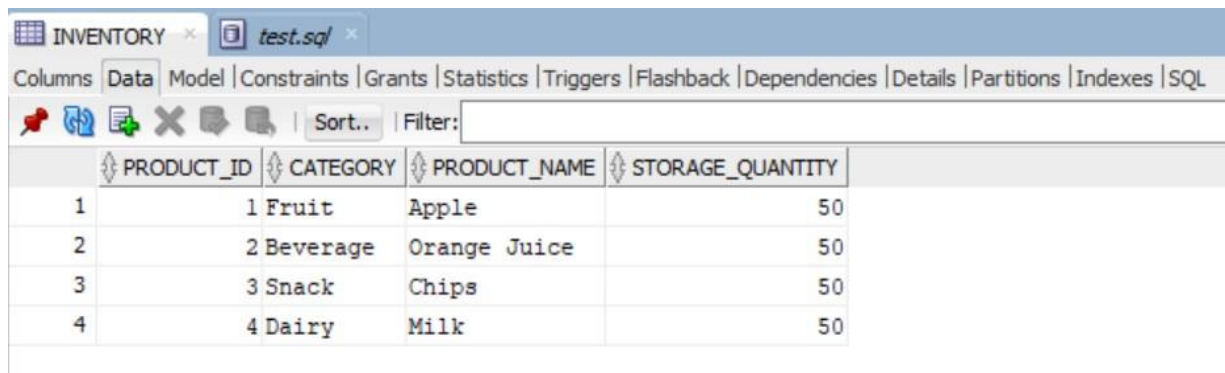
- c. The left hand side is just Transaction\_ID, which is already the sole candidate key and superkey in the relation. No further decomposition is needed.

4. Derive final schema

- c. We derive the final schema of  
Receipt(Transaction\_ID#, Product\_List, Points\_Earned)  
Transaction\_ID is the candidate key and determines all other attributes in the relation, therefore the relation satisfies BCNF and 3NF.



Inventory(Product\_ID#, Category, Product\_Name, Storage\_Quantity)



The screenshot shows a database management interface with a tab labeled 'INVENTORY' and a file named 'test.sql'. The 'Columns' tab is selected, displaying a table with four columns: PRODUCT\_ID, CATEGORY, PRODUCT\_NAME, and STORAGE\_QUANTITY. The table contains four rows of data.

	PRODUCT_ID	CATEGORY	PRODUCT_NAME	STORAGE_QUANTITY
1	1	Fruit	Apple	50
2	2	Beverage	Orange Juice	50
3	3	Snack	Chips	50
4	4	Dairy	Milk	50

Functional Dependencies:

$\{\text{Product\_ID}\} \rightarrow \text{Category}$   
 $\{\text{Product\_ID}\} \rightarrow \text{Product\_Name}$   
 $\{\text{Product\_ID}\} \rightarrow \text{Storage\_Quantity}$

Function Dependencies Removed  
Redundancy:

$\{\text{Product\_ID}\} \rightarrow \text{Storage\_Quantity}$

Bernstein's Algorithm:

1. Minimize the list of FDs ( $\text{FD } X \rightarrow Y, Z$ )
  - a. FD1:  $\{\text{Product\_ID}\} \rightarrow \text{Category}$
  - b. FD2:  $\{\text{Product\_ID}\} \rightarrow \text{Product\_Name}$

c. FD3:  $\{\text{Product\_ID}\} \rightarrow \text{Storage\_Quantity}$

d. Combine the FDs to form a single FD

i.  $\{\text{Product\_ID}\} \rightarrow \text{Category, Product\_Name, Storage\_Quantity}$

2. Get rid of redundant FDs

Find  $X^+$  in the reduced list, If  $X^+$  contains Y, then X, Y is redundant

a. We see that Category and Product\_Name are repeated FDs from the Product table, we eliminate these dependencies from the table.

c. This leaves  $\{\text{Product\_ID}\} \rightarrow \text{Storage\_Quantity}$

3. Minimize left hand side

a. The left hand side is just Product\_ID, which is already the sole candidate key and superkey in the relation. No further decomposition is needed.

4. Derive final schema

d. We derive the final schema of

Inventory(Product\_ID#, Storage\_Quantity)

Product\_ID is the candidate key and determines all other attributes in the relation, therefore the relation satisfies BCNF and 3NF.

The Inventory table relation is in **3NF**, all non-key values contained in the table are non-transitive and dependent on only the primary key.

