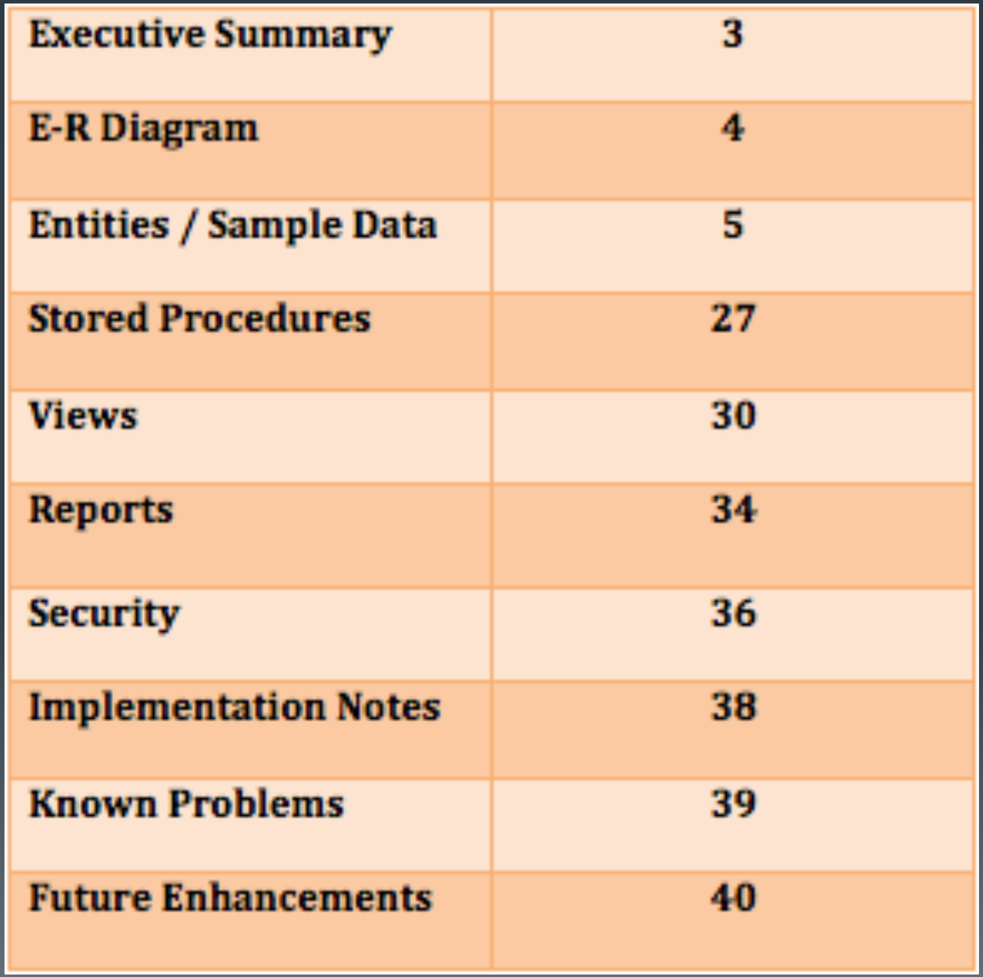
ND Sportswear

Nick DiCamillo
Database Management
Fall 2013

Table of Contents

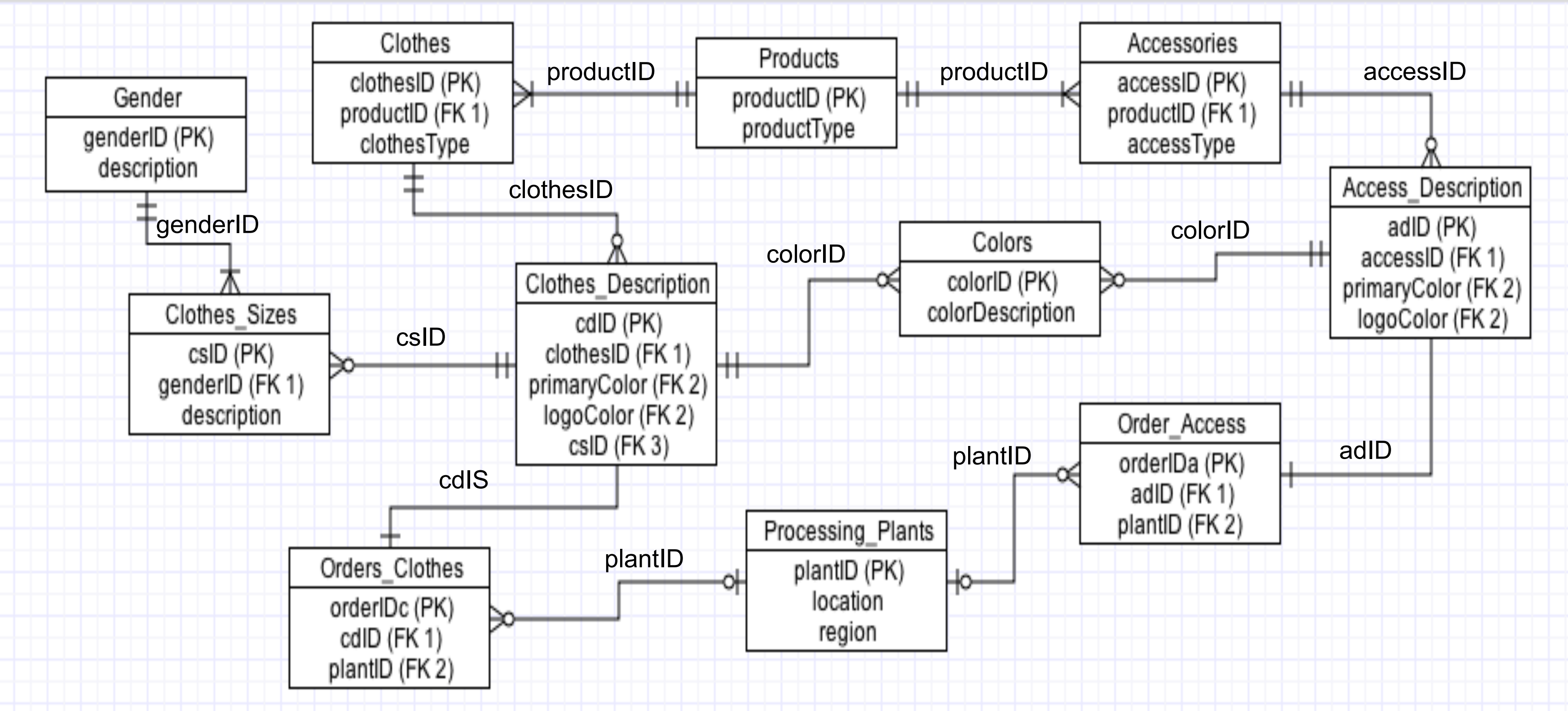


Executive Summary

- ND Sportswear is a new sportswear company that produces various Sportswear and Accessories for all athletes. Our design is simplistic yet stylish. All our products come in a single primary color with a ND logo, both in the color of your choice. We range from every age old or young with sizes that can fit any body type.

- This documents objective is to describe the database that is behind our manufacturing process. Here we see the order; what product it consists of, the size (if it is clothing), the primary color, and the logo color. We can also see what order it is and which processing plant it is being shipped to. The processing plants deal with the shipping and billing of the customer.

ER Diagram



Gender
- genderID (PK)
- description

Clothes
- clothesID (PK)
- productID (FK 1)
- clothesType

Products
- productID (PK)
- productType

Accessories
- accessID (PK)
- productID (FK 1)
- accessType

Access_Description
- adID (PK)
- accessID (FK 1)
- primaryColor (FK 2)
- logoColor (FK 2)

Clothes_Sizes
- csID (PK)
- genderID (FK 1)
- description

Clothes_Description
- cdID (PK)
- clothesID (FK 1)
- primaryColor (FK 2)
- logoColor (FK 2)
- csID (FK 3)

Colors
- colorID (PK)
- colorDescription

Order_Access
- orderIDa (PK)
- adID (FK 1)
- plantID (FK 2)

Processing_Plants
- plantID (PK)
- location
- region

Orders_Clothes
- orderIDc (PK)
- cdID (FK 1)
- plantID (FK 2)

Products

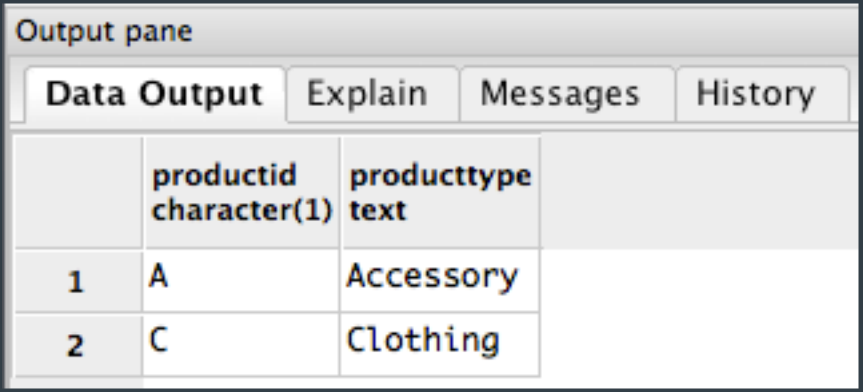
- Purpose:
  - Our company sells two types of products, Clothes and Accessories. Clothes consist of shirts, pants, sweats and compressions. Accessories consists of sweatbands and bags. To differentiate between the two, products under clothing are labeled with a C and accessories are labeled A.

- SQL Create Statement:

```
CREATE TABLE Products(
            productID        char(1) not null,
            productType      text,
primary key(productID)
);
```

- Functional Dependencies
  - productID -> productType

Products Sample Data



	productid character(1)	producttype text
1	A	Accessory
2	C	Clothing

Clothes

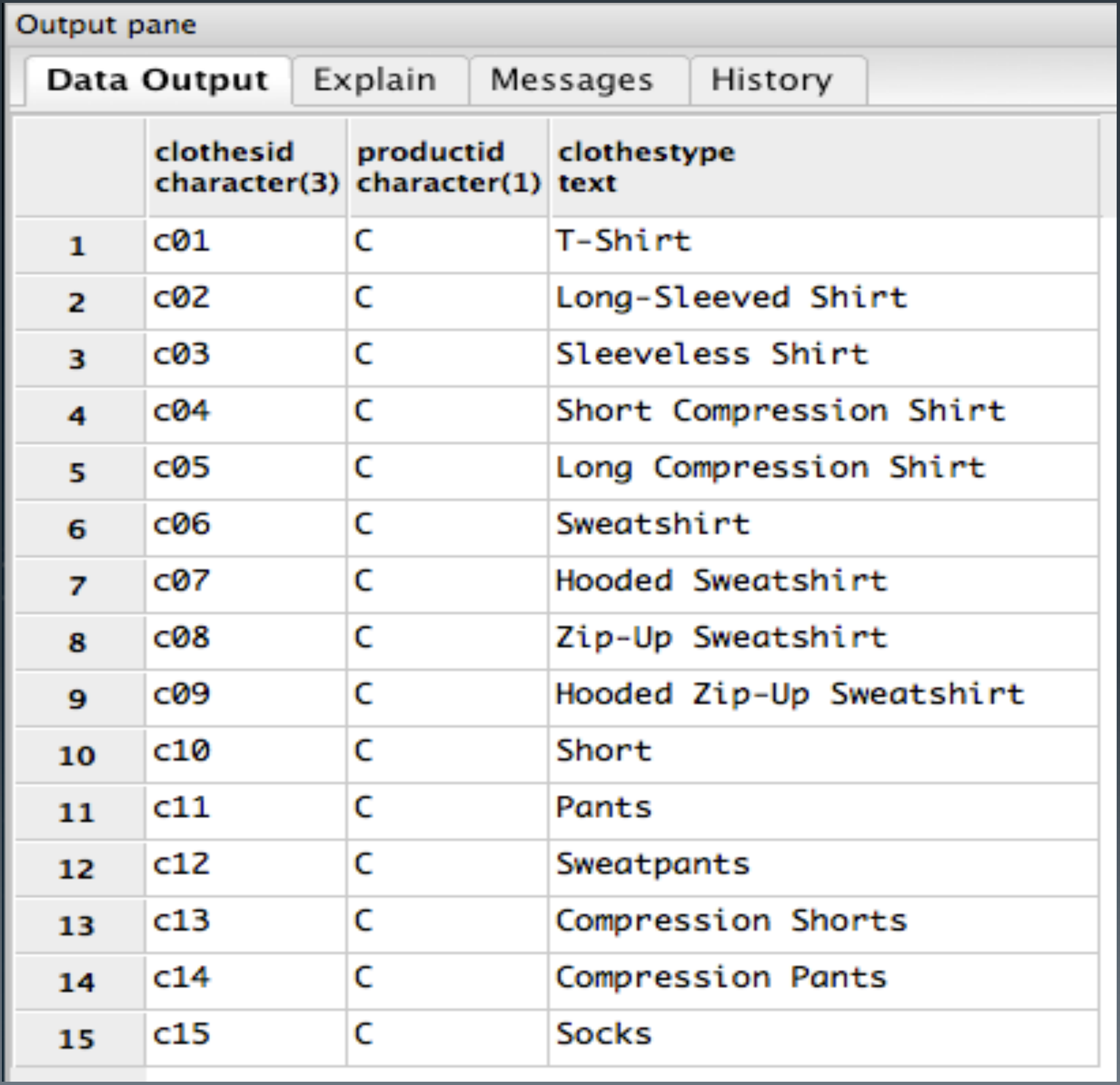
- Purpose:
  - Displays all the different types of clothing we offer

- SQL Create Statement:

```
CREATE TABLE Clothes(
        clothesID       char(3) not null,
        productID       char(1) not null references Products(productID),
        clothesType     text,
primary key(clothesID)
);
```

- Functional Dependencies
  - clothesID -> productID, clothesType

Clothes Sample Data



	clothesid character(3)	productid character(1)	clothestype text
1	c01	C	T-Shirt
2	c02	C	Long-Sleeved Shirt
3	c03	C	Sleeveless Shirt
4	c04	C	Short Compression Shirt
5	c05	C	Long Compression Shirt
6	c06	C	Sweatshirt
7	c07	C	Hooded Sweatshirt
8	c08	C	Zip-Up Sweatshirt
9	c09	C	Hooded Zip-Up Sweatshirt
10	c10	C	Short
11	c11	C	Pants
12	c12	C	Sweatpants
13	c13	C	Compression Shorts
14	c14	C	Compression Pants
15	c15	C	Socks

Gender

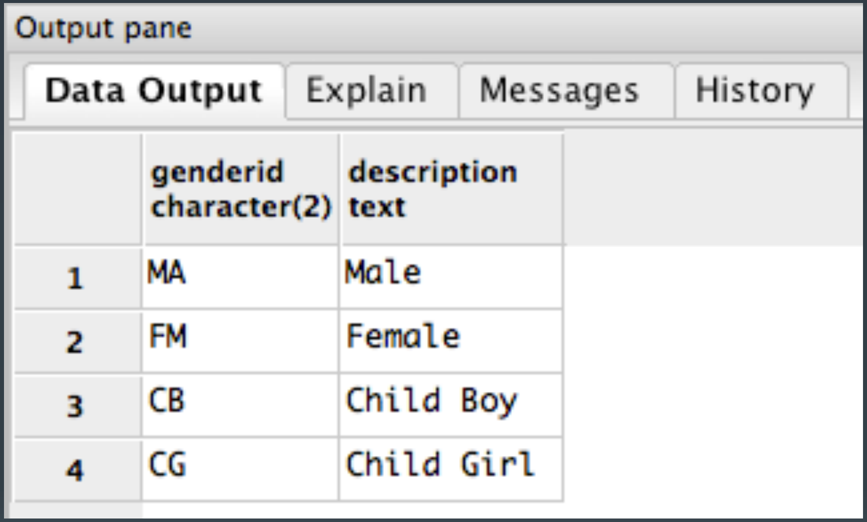
- Purpose
  - Will be used to differentiates different sizes among genders

- SQL Create Statement:

```
CREATE TABLE Gender(
        genderID        char(2) not null,
        description     text,
primary key(genderID)
);
```

- Functional Dependencies
  - genderID -> description

Gender Sample Data



	genderid character(2)	description text
1	MA	Male
2	FM	Female
3	CB	Child Boy
4	CG	Child Girl

Clothes_Size

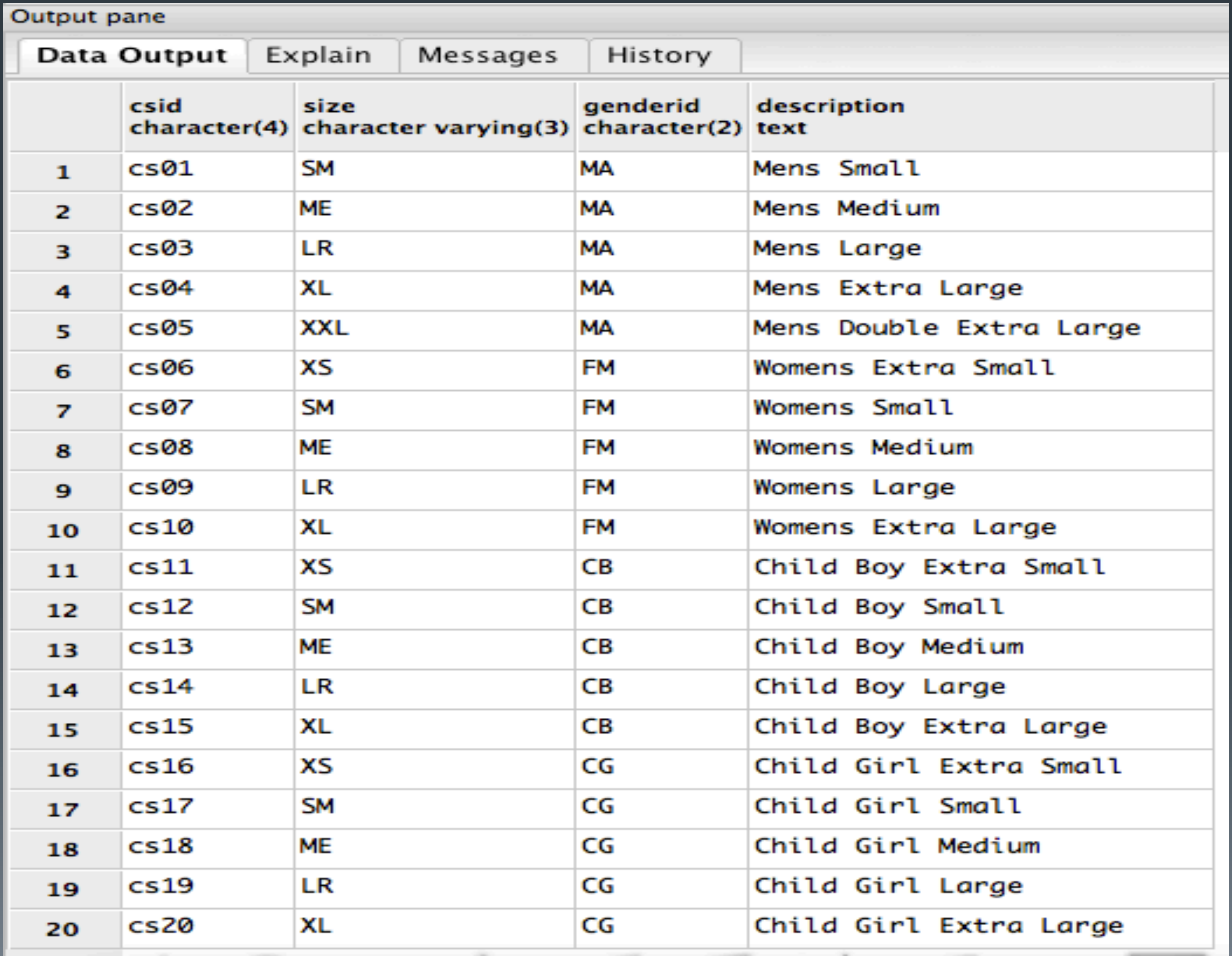
- Purpose
  - Displays different sizes among genders

- SQL Create Statement:

```
CREATE TABLE Clothes_Size(
          csID            char(4) not null,
          size            varchar(3),
          genderID        char(2) references Gender(genderID),
          description     text,
primary key(csID)
);
```

- Functional Dependencies
  - csID -> size, genderID, description

Clothes_Size Sample Data



	csid character(4)	size character varying(3)	genderid character(2)	description text
1	cs01	SM	MA	Mens Small
2	cs02	ME	MA	Mens Medium
3	cs03	LR	MA	Mens Large
4	cs04	XL	MA	Mens Extra Large
5	cs05	XXL	MA	Mens Double Extra Large
6	cs06	XS	FM	Womens Extra Small
7	cs07	SM	FM	Womens Small
8	cs08	ME	FM	Womens Medium
9	cs09	LR	FM	Womens Large
10	cs10	XL	FM	Womens Extra Large
11	cs11	XS	CB	Child Boy Extra Small
12	cs12	SM	CB	Child Boy Small
13	cs13	ME	CB	Child Boy Medium
14	cs14	LR	CB	Child Boy Large
15	cs15	XL	CB	Child Boy Extra Large
16	cs16	XS	CG	Child Girl Extra Small
17	cs17	SM	CG	Child Girl Small
18	cs18	ME	CG	Child Girl Medium
19	cs19	LR	CG	Child Girl Large
20	cs20	XL	CG	Child Girl Extra Large

Accessories

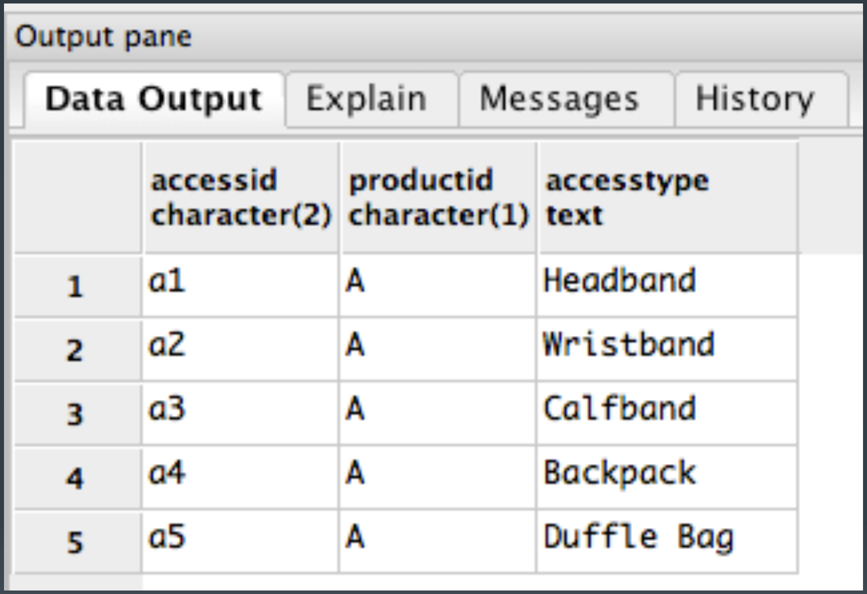
- Purpose
  - Displays different accessories that we offer

- SQL Create Statement:

```
CREATE TABLE Accessories(
          accessID        char(2) not null,
          productID       char(1) not null references Products(productID),
          accessType      text,
primary key(accessID)
);
```

- Functional Dependencies
  - accessID -> productID, accessType

Accessory Sample Data



	accessid character(2)	productid character(1)	accesstype text
1	a1	A	Headband
2	a2	A	Wristband
3	a3	A	Calfband
4	a4	A	Backpack
5	a5	A	Duffle Bag

Colors

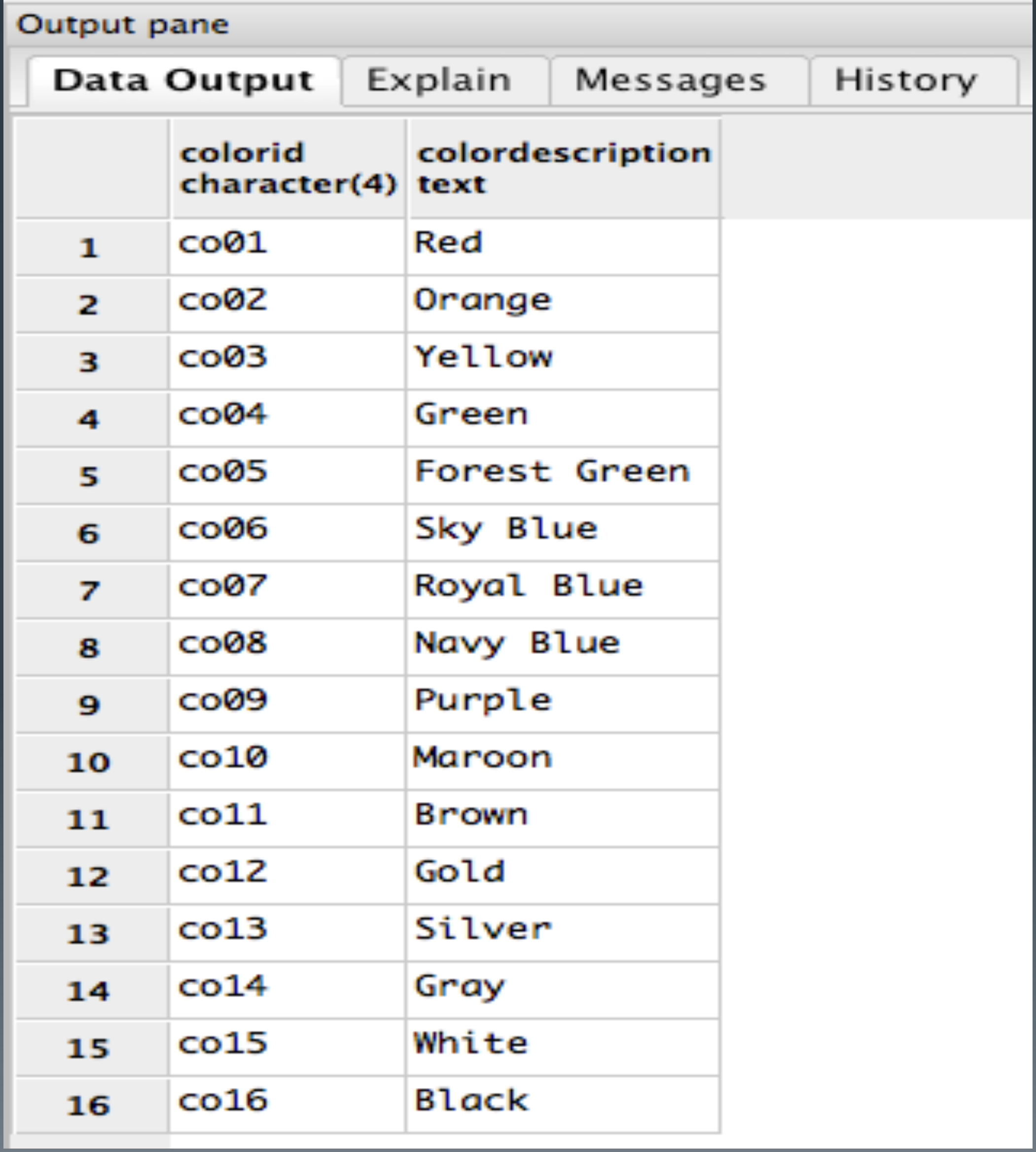
- Purpose
  - The colors our products come in

- SQL Create Statement

```
CREATE TABLE Colors(
        colorID                    char(4) not null,
        colorDescription           text,
primary key(colorID)
);
```

- Functional Dependencies
  - colorID -> colorDescription

Color Sample Data



	colorid character(4)	colordescription text
1	co01	Red
2	co02	Orange
3	co03	Yellow
4	co04	Green
5	co05	Forest Green
6	co06	Sky Blue
7	co07	Royal Blue
8	co08	Navy Blue
9	co09	Purple
10	co10	Maroon
11	co11	Brown
12	co12	Gold
13	co13	Silver
14	co14	Gray
15	co15	White
16	co16	Black

Clothes_Description

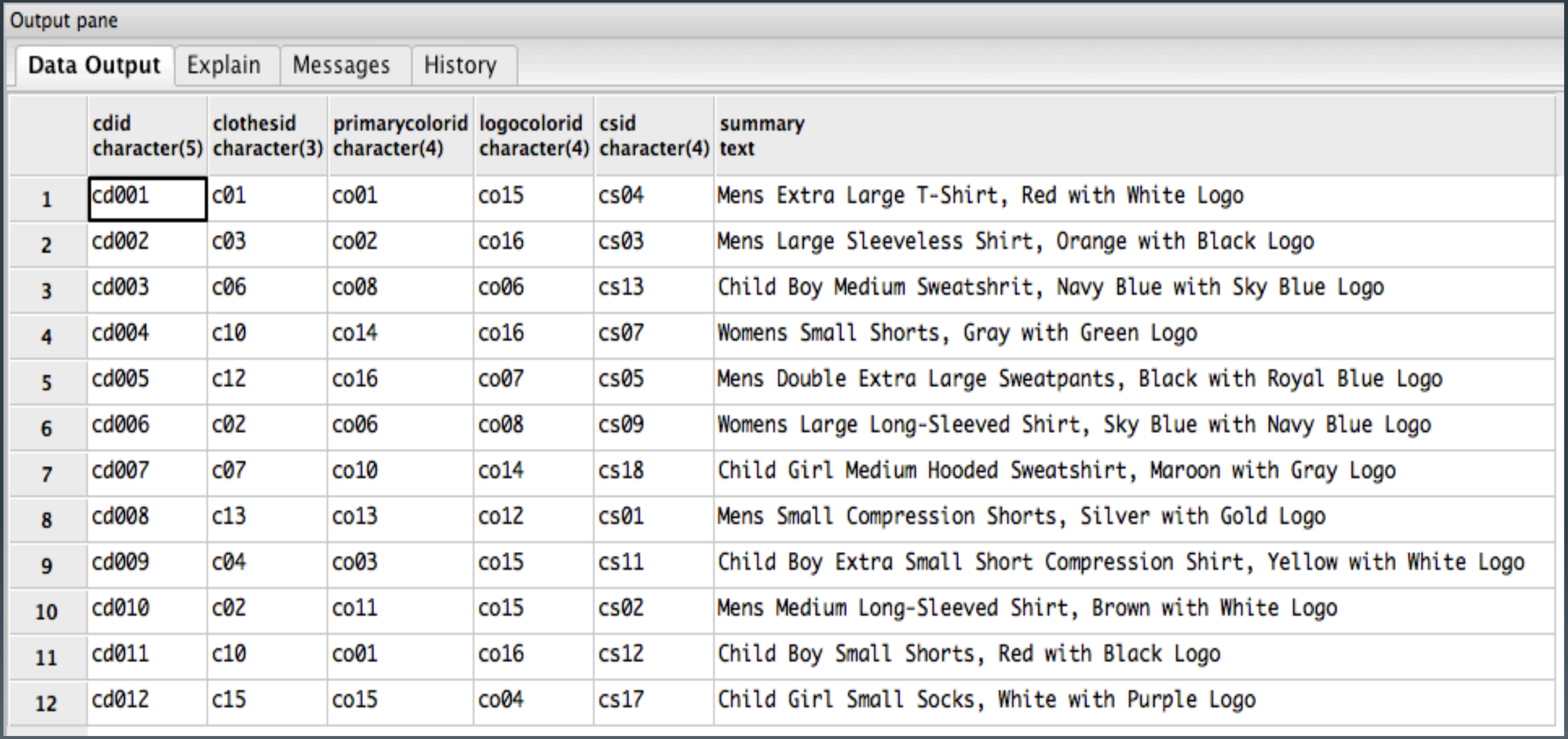
- Purpose
  - Describes a customer order

- SQL Create Statement

```
CREATE TABLE Clothes_Description(
          cdID              char(5) not null,
          clothesID         char(3) references Clothes(clothesID),
          primaryColorID    char(4) references Colors(colorID),
          logoColorID       char(4) references Colors(colorID),
          csID              char(4) references Clothes_Size(csID),
          summary           text,
primary key(cdID)
);
```

- Functional Dependencies
  - cdID -> clothesID, primaryColorID, logoColorID, csID, summary

Clothes_Description Sample Data



	cdid character(5)	clothesid character(3)	primarycolorid character(4)	logocolorid character(4)	csid character(4)	summary text
1	cd001	c01	co01	co15	cs04	Mens Extra Large T-Shirt, Red with White Logo
2	cd002	c03	co02	co16	cs03	Mens Large Sleeveless Shirt, Orange with Black Logo
3	cd003	c06	co08	co06	cs13	Child Boy Medium Sweatshrit, Navy Blue with Sky Blue Logo
4	cd004	c10	co14	co16	cs07	Womens Small Shorts, Gray with Green Logo
5	cd005	c12	co16	co07	cs05	Mens Double Extra Large Sweatpants, Black with Royal Blue Logo
6	cd006	c02	co06	co08	cs09	Womens Large Long-Sleeved Shirt, Sky Blue with Navy Blue Logo
7	cd007	c07	co10	co14	cs18	Child Girl Medium Hooded Sweatshirt, Maroon with Gray Logo
8	cd008	c13	co13	co12	cs01	Mens Small Compression Shorts, Silver with Gold Logo
9	cd009	c04	co03	co15	cs11	Child Boy Extra Small Short Compression Shirt, Yellow with White Logo
10	cd010	c02	co11	co15	cs02	Mens Medium Long-Sleeved Shirt, Brown with White Logo
11	cd011	c10	co01	co16	cs12	Child Boy Small Shorts, Red with Black Logo
12	cd012	c15	co15	co04	cs17	Child Girl Small Socks, White with Purple Logo

Access_Description

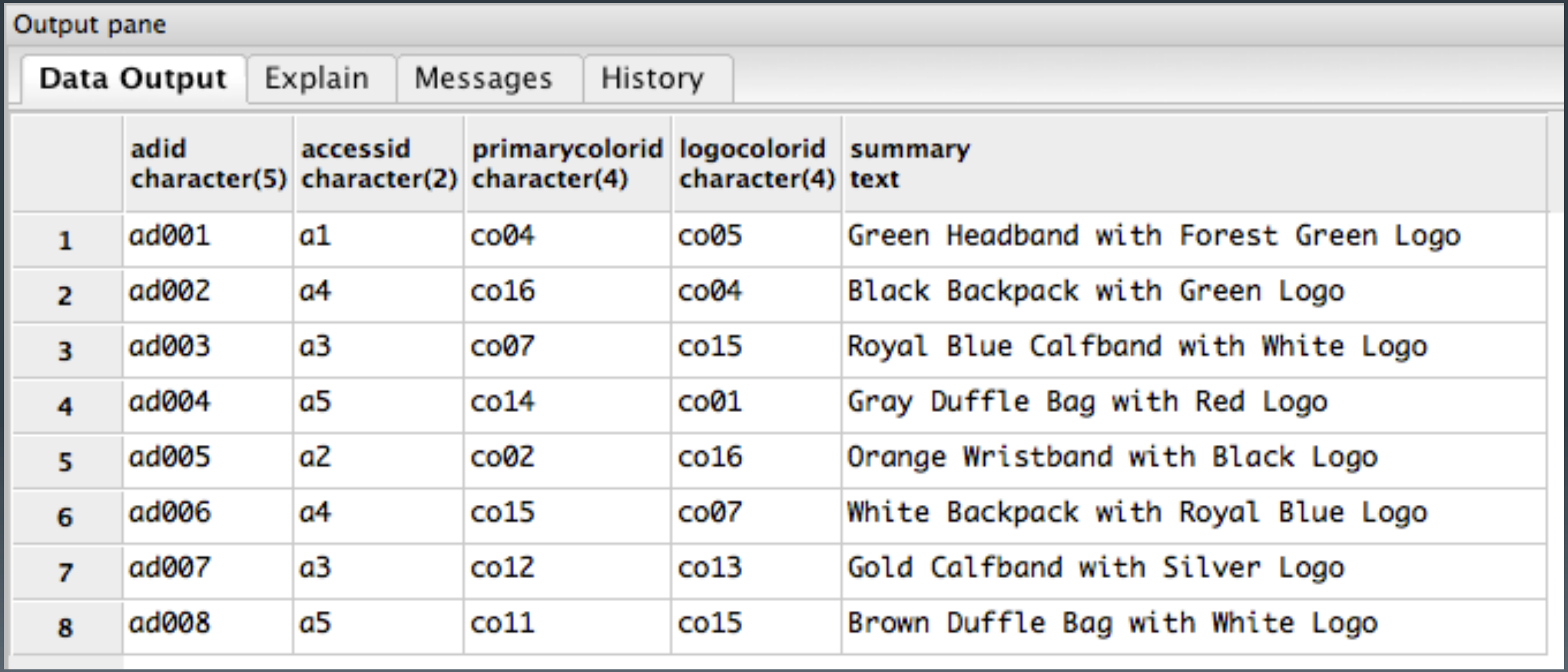
- Purpose:
  - Describes an accessory order

- SQL Create Statement:

```
CREATE TABLE Access_Description(
        adID            char(5) not null,
        accessID        char(2) references Accessories(accessID),
        primaryColorID  char(4) references Colors(colorID),
        logoColorID     char(4) references Colors(colorID),
        summary         text,
primary key(adID)
);
```

- Functional Dependencies
  - adID -> accessID, primaryColorID, logoColorID, summary

Access_Description Sample Data



	adid character(5)	accessid character(2)	primarycolorid character(4)	logocolorid character(4)	summary text
1	ad001	a1	co04	co05	Green Headband with Forest Green Logo
2	ad002	a4	co16	co04	Black Backpack with Green Logo
3	ad003	a3	co07	co15	Royal Blue Calfband with White Logo
4	ad004	a5	co14	co01	Gray Duffle Bag with Red Logo
5	ad005	a2	co02	co16	Orange Wristband with Black Logo
6	ad006	a4	co15	co07	White Backpack with Royal Blue Logo
7	ad007	a3	co12	co13	Gold Calfband with Silver Logo
8	ad008	a5	co11	co15	Brown Duffle Bag with White Logo

Processing_Plants

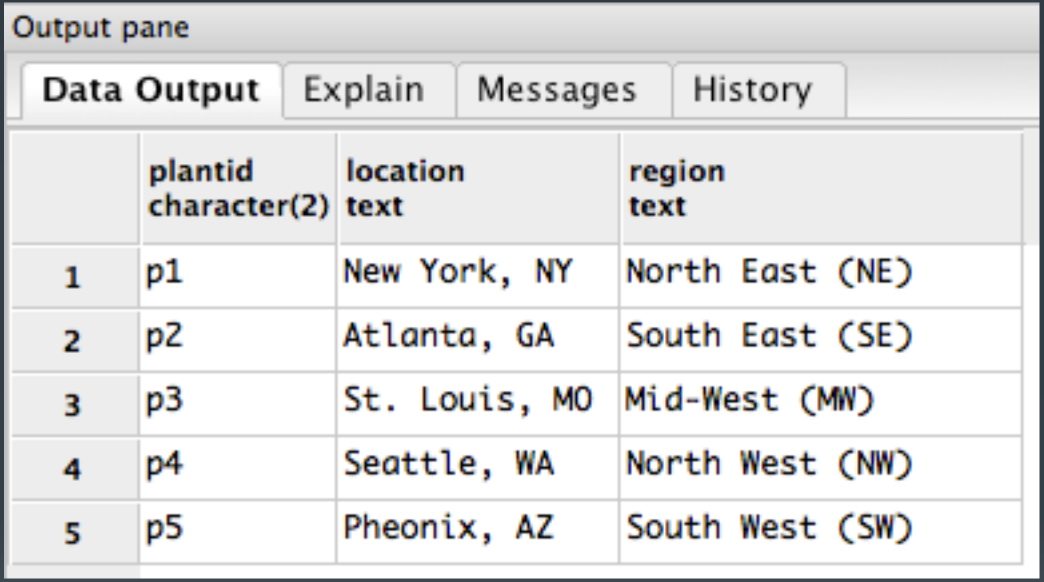
- Purpose
  - Different processing plants and the regions of the country they cover

- SQL Create Statement

```
CREATE TABLE Processing_Plants(
        plantID         char(2) not null,
        location        text,
        region          char(2),
primary key(plantID)
);
```

- Functional Dependencies
  - plantID -> location, region

Processing_Plants Sample Data



	plantid character(2)	location text	region text
1	p1	New York, NY	North East (NE)
2	p2	Atlanta, GA	South East (SE)
3	p3	St. Louis, MO	Mid-West (MW)
4	p4	Seattle, WA	North West (NW)
5	p5	Pheonix, AZ	South West (SW)

Orders_Clothes

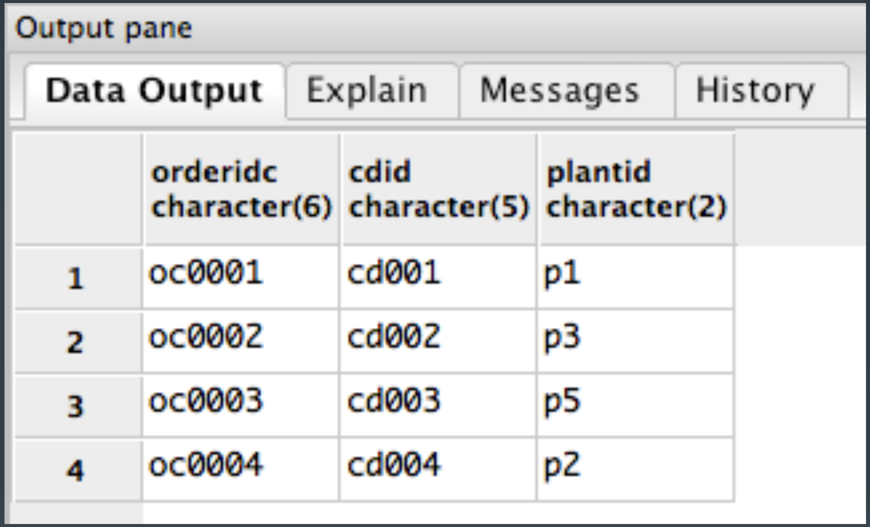
- Purpose
  - The order number for a clothing order with the ID of the description and the processing plant it is to be shipped to

- SQL Create Statement

```
CREATE TABLE Orders_Clothes(
        orderIDc        char(6) not null,
        cdID            char(5) references Clothes_Description(cdID),
        plantID         char(2) references Processing_Plants(plantID),
primary key(orderIDc)
);
```

- Functional Dependencies
  - orderIDc -> cdID, plantID

Orders_Clothes Sample Data



	orderidc character(6)	cdid character(5)	plantid character(2)
1	oc0001	cd001	p1
2	oc0002	cd002	p3
3	oc0003	cd003	p5
4	oc0004	cd004	p2

Order_Access

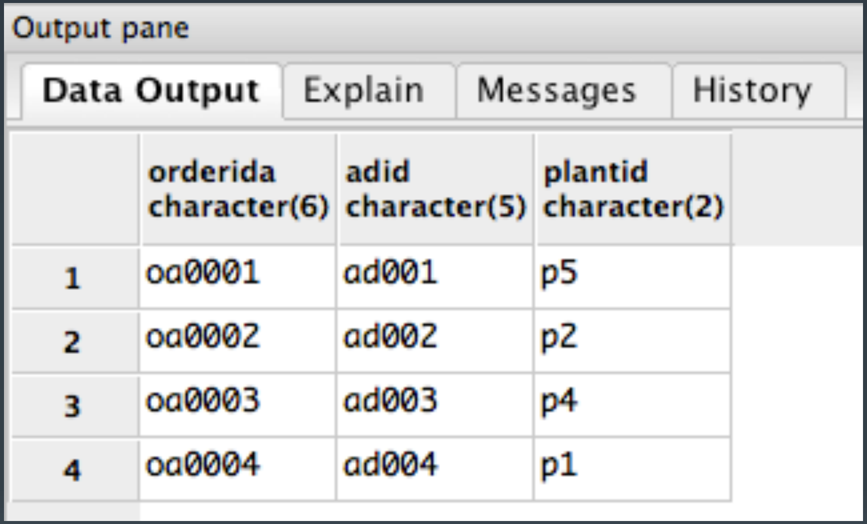
- Purpose:
  - The order number for an accessory order with the ID of the description and the processing plant it is to be shipped to

- SQL Create Statement

```
CREATE TABLE Orders_Access(
        orderIDa        char(6) not null,
        adID            char(5) references Access_Description(adID),
        plantID         char(2) references Processing_Plants(plantID),
primary key(orderIDa)
);
```

- Functional Dependencies
  - orderIDa -> adID, plantID

Orders_Access Sample Data



	orderida character(6)	adid character(5)	plantid character(2)
1	oa0001	ad001	p5
2	oa0002	ad002	p2
3	oa0003	ad003	p4
4	oa0004	ad004	p1

Stored Procedure Size

- To verify the customer has selected a size in the order description

```
CREATE FUNCTION Size_Selected()
RETURNS trigger AS $$
DECLARE
        csID INTEGER := 0;
BEGIN
        IF Clothes_Order.csID IS NULL THEN
        RAISE EXCEPTION 'No Size Selected';
        END IF;
END;
$$ LANGUAGE plpgsql;
```

Stored Procedure Color

- To verify the customer has selected a primary color and a logo color for their order

```
CREATE FUNCTION Colors_Selected()
RETURNS trigger AS $$
DECLARE
        primaryColor INTEGER := 0;
        logoColor INTEGER := 0;
BEGIN
        IF Clothes_Order.primaryColorID IS NULL THEN
                RAISE EXCEPTION 'No Primary Color Selected';
        END IF;

        IF Clothes_Order.logoColorID IS NULL THEN
                RAISE EXCEPTION 'No Logo Color Selected';
        END IF;
```

Stored Procedure Color (cont’d)

```
        IF Access_Order.primaryColorID IS NULL THEN
                RAISE EXCEPTION 'No Primary Color Selected';
        END IF;

        IF Access_Order.logoColorID IS NULL THEN
                RAISE EXCEPTION 'No Logo Color Selected';
        END IF;
END;
$$ LANGUAGE plpgsql;
```

Men’s Clothing View

```
CREATE VIEW MaleClothing AS
SELECT DISTINCT Clothes.clothesID AS Clothing_ID,
                Clothes_Size.csID AS Size_ID,
                Clothes.clothesType AS Clothing,
                Clothes_Size.size AS Size
FROM Clothes, Clothes_Size, Gender
WHERE Clothes_Size.genderID = ’MA’
ORDER BY Clothes.clothesID, Clothes_Size.csID;
```

Women’s Clothing View

```
CREATE VIEW FemaleClothing AS
SELECT DISTINCT Clothes.clothesID AS Clothing_ID,
                Clothes_Size.csID AS Size_ID,
                Clothes.clothesType AS Clothing,
                Clothes_Size.size AS Size
FROM Clothes, Clothes_Size, Gender
WHERE Clothes_Size.genderID = 'FM’
ORDER BY Clothes.clothesID, Clothes_Size.csID;
```

Child Boy Clothing View

```
CREATE VIEW ChildBoyClothing AS
SELECT DISTINCT Clothes.clothesID AS Clothing_ID,
                Clothes_Size.csID AS Size_ID,
                Clothes.clothesType AS Clothing,
                Clothes_Size.size AS Size
FROM Clothes, Clothes_Size, Gender
WHERE Clothes_Size.genderID = ’CB’
ORDER BY Clothes.clothesID, Clothes_Size.csID;
```

Child Girl Clothing View

```
CREATE VIEW ChildGirlClothing AS
SELECT DISTINCT Clothes.clothesID AS Clothing_ID,
                Clothes_Size.csID AS Size_ID,
                Clothes.clothesType AS Clothing,
                Clothes_Size.size AS Size
FROM Clothes, Clothes_Size, Gender
WHERE Clothes_Size.genderID = ’CG’
ORDER BY Clothes.clothesID, Clothes_Size.csID;
```

Most Popular Clothing

- Report to display the most popular clothing product purchased

```
SELECT Clothes_Description.clothesID,
             Clothes.clothesType
FROM Clothes, Clothes_Description
WHERE Clothes.clothesID = Clothes_Description.clothesID
ORDER BY Clothes.clothesID
```

Most Popular Accessory

- Report to display the most popular accessory purchased

```
SELECT Access_Description.accessID,
            Accessories.accessType
FROM Accessories, Access_Description
WHERE Accessories.accessID = Access_Description.accessID
ORDER BY Accessories.accessID
```

Security

```
CREATE ROLE NDadmin
GRANT SELECT, INSERT, UPDATE
ON ALL TABLES IN SCHEMA PUBLIC TO NDadmin
```

- Allows the admin to update and view all tables

Security

```
CREATE ROLE NDcustomer

GRANT SELECT, INSERT, UPDATE
ON Clothes_Description, Access_Description TO NDcustomer

GRANT SELECT
ON Clothes, Accessories, Clothes_Sizes, Colors TO Ndcustomer
```

- Allows a customer to view and update their order description
- Allows a customer to view clothes and accessories, sizes, and colors

Implementation Notes

- This database should be able to connect to the database implemented at our processing plants. This will enable us to be able to track manufacturing and processing together. Here we can give customers a better timetable of arrival of their product and they will also be able to track their order.

- Prices of our products will be based on a supply/demand principle.

Known Problems

- Simple views
  - Complex views would increase productivity

- No way to track order progress for customer

Future Enhancements

- View to show all different color possibilities with all products
- Stored procedure to verify items are in stock
- Create reports to see the processing plants with the most orders
- Create reports to see which is the most popular Primary and Logo color of choice