```
-- ************************************************************

-- This script creates the database
-- Safa Alasady
-- for CIS 3050-xx Project 2
-- Prof. Ahmed Azam
-- March 31, 2024

-- ************************************************************

DROP TABLE IF EXISTS customers;
DROP TABLE IF EXISTS orders;
DROP TABLE IF EXISTS order_details;
DROP TABLE IF EXISTS items;
DROP TABLE IF EXISTS artists;
DROP TABLE IF EXISTS employees;

-- create tables

CREATE TABLE customers

(

customer_id          INT          ,

customer_first_name   VARCHAR(20),

customer_last_name    VARCHAR(20)     NOT NULL,
```

```
customer_address      VARCHAR(50)    NOT NULL,

customer_city         VARCHAR(20)     NOT NULL,

customer_state        CHAR(2)         NOT NULL,

customer_zip          CHAR(5)     NOT NULL,

customer_phone        CHAR(10)     NOT NULL,

customer_fax          CHAR(10),

CONSTRAINT customers_pk
  PRIMARY KEY (customer_id)

)
;



CREATE TABLE artists

(

artist_id          INT             NOT NULL,

artist_name   VARCHAR(30),

CONSTRAINT artist_pk
```

```
  PRIMARY KEY (artist_id)

)
;


CREATE TABLE items

(

item_id       INT             NOT NULL,

title         VARCHAR(50)     NOT NULL,

artist_id     INT     NOT NULL,

unit_price    DECIMAL(9,2)    NOT NULL,

CONSTRAINT items_pk
    PRIMARY KEY (item_id),

CONSTRAINT items_fk_artists
    FOREIGN KEY (artist_id) REFERENCES artists (artist_id)
);



CREATE TABLE employees
```

```
(

employee_id         INT              NOT NULL,

last_name           VARCHAR(20)      NOT NULL,

first_name          VARCHAR(20)      NOT NULL,

manager_id          INT

,
CONSTRAINT employees_pk
  PRIMARY KEY (employee_id),
CONSTRAINT emp_fk_mgr FOREIGN KEY (manager_id) REFERENCES
employees(employee_id) );



CREATE TABLE orders

(

order_id          INT         NOT NULL,

customer_id       INT         NOT NULL,

order_date        DATE        NOT NULL,

shipped_date      DATE,
```

```
employee_id       INT,

CONSTRAINT orders_pk
   PRIMARY KEY (order_id),

CONSTRAINT orders_fk_customers
   FOREIGN KEY (customer_id) REFERENCES customers (customer_id),
CONSTRAINT orders_fk_employees
   FOREIGN KEY (employee_id) REFERENCES employees (employee_id)

);



CREATE TABLE order_details

(

order_id      INT          NOT NULL,

item_id       INT          NOT NULL,

order_qty     INT          NOT NULL,

CONSTRAINT order_details_pk
   PRIMARY KEY (order_id, item_id),

CONSTRAINT order_details_fk_orders
   FOREIGN KEY (order_id)
```

```
    REFERENCES orders (order_id),

CONSTRAINT order_details_fk_items
    FOREIGN KEY (item_id)
    REFERENCES items (item_id)

);




-- insert rows into tables

INSERT INTO customers VALUES

(1,'Oliva','Jachson','1555 W Lane Ave','Columbus','OH','43221','6145554435','6145553928'),

(2,'Emma','Randall','11 E Rancho Madera
Rd','Madison','WI','53707','2095551205','2095552262'),

(3,'Johnathon','Millerton','60 Madison Ave','New York','NY','10010','2125554800',NULL),

(4,'Charlotte','Mayte','2021 K Street Nw','Washington','DC','20006','2025555561',NULL),

(5,'Kendall','Davis','4775 E Miami River Rd','Cleves','OH','45002','5135553043',NULL),

(6,'Kaitlin','Brown','3250 Spring Grove Ave','Cincinnati','OH','45225','8005551957','8005552826'),
```

(7,'Lily','Chaddick','9022 E Merchant Wy','Fairfield','IA','52556','5155556130',NULL),

(8,'Deborah','Davis','415 E Olive Ave','Fresno','CA','93728','5595558060',NULL),

(9,'Karina','Miller','882 W Easton Wy','Los Angeles','CA','90084','8005557000',NULL),

(10,'Kurt','Nickalus','28210 N Avenue
Stanford','Valencia','CA','91355','8055550584','055556689'),

(11,'Hinrey','Lopez','7833 N Ridge Rd','Sacramento','CA','95887','2095557500','2095551302'),

(12,'Anders','Rohansen','12345 E 67th Ave NW','Takoma
Park','MD','24512','3385556772',NULL),

(13,'','Neftaly','2508 W Shaw Ave','Fresno','CA','93711','5595556245',NULL),

(14,'Gonzalo','Keeton','12 Daniel Road','Fairfield','NJ','07004','2015559742',NULL),

(15,'Ania','Irvin','1099 N Farcourt St','Orange','CA','92807','7145559000',NULL),

(16,'Dakota','Baylee','1033 N Sycamore Ave.','Los Angeles','CA','90038','2135554322',NULL),

(17,'Samuel','Jacobsen','3433 E Widget Ave','Palo Alto','CA','92711','4155553434',NULL),

(18,'Justin','Javen','828 S Broadway','Tarrytown','NY','10591','8005550037',NULL),

(19,'Kyle','Marissa','789 E Mercy Ave','Phoenix','AZ','85038','9475553900',NULL),

(20,'Mohammad','Ali','Five Lakepointe Plaza, Ste
500','Charlotte','NC','28217','7045553500',NULL),

(21,'Marvin','Quintin','2677 Industrial Circle
Dr','Columbus','OH','43260','6145558600','6145557580'),

(22,'Rashad','Holbrooke','3467 W Shaw Ave
#103','Fresno','CA','93711','5595558625','5595558495'),

(23,'Theo','Hernandez','627 Aviation Way','Manhatttan Beach','CA','90266','3105552732',NULL),

(24,'Julian','Carson','372 San Quentin','San Francisco','CA','94161','6175550700',NULL),

(25,'Kirsten','Story','2401 Wisconsin Ave NW','Washington','DC','20559','2065559115',NULL),

(26,'Ahmed','Azam','372 San Quentin','San Francisco','CA','94161','6175550700',NULL);

INSERT INTO artists(artist_id,artist_name) VALUES

(10, 'Umani'),

(11, 'The Ubernerds'),

(12, 'No Rest For The Weary'),

(13, 'Burt Ruggles'),

(14, 'Sewed the Vest Pocket'),

(15, 'Jess & Odie'),

(16, 'Onn & Onn');

INSERT INTO items (item_id,title,artist_id,unit_price) VALUES

(1,'Umami In Concert',10,17.95),

(2,'Race Car Sounds',11,13),

(3,'No Rest For The Weary',12,16.95),

(4,'More Songs About Structures and Comestibles',12,17.95),

(5,'On The Road With Burt Ruggles',13,17.5),

(6,'No Fixed Address',14,16.95),

(7,'Rude Noises',15,13),

(8,'Burt Ruggles: An Intimate Portrait',13,17.95),

(9,'Zone Out With Umami',10,16.95),

(10,'Etcetera',16,17);

INSERT INTO employees VALUES

(1,'Garcia', 'Mia', null),

(2,'Moore', 'Aria', 1),

(3,'Lee', 'Layla', 2),

(9,'Locario', 'Paulo',1),

(8,'Leary', 'Rhea',9),

(4,'Hernandez','Olivia',9),

(5,'Aaronsen', 'Robert',4),

(6,'White', 'Asher',8),

(7,'clark', 'Thomas',2);

INSERT INTO orders VALUES

(19, 1, '2012-10-23', '2012-10-28', 6),

(29, 8, '2012-11-05', '2012-11-11', 6),

(32, 11, '2012-11-10', '2012-11-13', NULL),

(45, 2, '2012-11-25', '2012-11-30', NULL),

(70, 10, '2012-12-28', '2013-01-07', 5),

(89, 22, '2013-01-20', '2013-01-22', 7),

(97, 20, '2013-01-29', '2013-02-02', 5),

(118, 3, '2013-02-24', '2013-02-28', 7),

(144, 17, '2013-03-21', '2013-03-29', NULL),

(158, 9, '2013-04-04', '2013-04-20', NULL),

(165, 14, '2013-04-11', '2013-04-13', NULL),

(180, 24, '2013-04-25', '2013-05-30', NULL),

(231, 15, '2013-06-14', '2013-06-22', NULL),

(242, 23, '2013-06-24', '2013-07-06', 3),

(264, 9, '2013-07-15', '2013-07-18', 6),

(298, 18, '2013-08-18', '2013-09-22', 3),

(321, 2, '2013-09-09', '2013-10-05', 6),

(381, 7, '2013-11-08', '2013-11-16', 7),
(413, 17, '2013-12-05', '2014-01-11', 7),

(442, 5, '2013-12-28', '2014-01-03', 5),

(479, 1, '2014-01-30', '2014-03-03', 3),

(491, 16, '2014-02-08', '2014-02-14', 5),

(523, 3, '2014-03-07', '2014-03-15', 3),

(548, 2, '2014-03-22', '2014-04-18', NULL),

(550, 17, '2014-03-23', '2014-04-03', NULL),

(601, 16, '2014-04-21', '2014-04-27', NULL),

(607, 20, '2014-04-25', '2014-05-04', NULL),

(624, 2, '2014-05-04', '2014-05-09', NULL),

(627, 17, '2014-05-05', '2014-05-10', NULL),

(630, 20, '2014-05-08', '2014-05-18', 7),

(651, 12, '2014-05-19', '2014-06-02', 7),

(658, 12, '2014-05-23', '2014-06-02', 7),

(687, 17, '2014-06-05', '2014-06-08', NULL),

(693, 9, '2014-06-07', '2014-06-19', NULL),

(703, 19, '2014-06-12', '2014-06-19', 7),

(778, 13, '2014-07-12', '2014-07-21', 7),

(796, 17, '2023-07-19', '2014-07-26', 5),

(800, 19, '2023-07-21', '2014-07-28', NULL),

(802, 2, '2023-07-21', '2014-07-31', NULL),

(824, 1, '2024-08-01', NULL, NULL),

(827, 18, '2024-08-02', NULL, NULL),

(829, 9, '2024-08-02', NULL, NULL);

INSERT INTO order_details VALUES

(381,1,1),
(601,9,1),
(442,1,1),
(523,9,1),

(630,5,1),
(778,1,1),
(693,10,1),
(118,1,1),

(264,7,1),
(607,10,1),
(624,7,1),
(658,1,1),

(800,5,1),

(158,3,1),

(321,10,1),

(687,6,1),

(827,6,1),

(144,3,1),

(479,1,2),

(630,6,2),

(796,5,1),

(97,4,1),

(601,5,1),

(800,1,1),

(29,10,1),

(70,1,1),

(165,4,1),

(180,4,1),

(231,10,1),

(413,10,1),

(491,6,1),

(607,3,1),

(651,3,1),
(703,4,1),

(802,3,1),
(824,7,2),

(829,1,1),
(550,4,1),
(796,7,1),
(693,6,1),
(29,3,1),
(32,7,1),
(242,1,1),

(298,1,1),
(479,4,1),
(548,9,1),
(627,9,1),

(778,3,1),

(19,5,1),
(89,4,1),

(242,6,1),
(264,4,1),
(550,1,1),

(693,7,3),

(824,3,1),

(829,5,1),

(829,9,1);

/* 1. Write a query that displays a list of all customers showing the customer's last name,

customer state, and phone number. Sort the results by customer state, then customer last name.

I THINK ITS DONE NOW

*/

SELECT customer_last_name, customer_state, customer_phone

FROM customers

ORDER BY customer_state, customer_last_name;

/* 2. Write a query that displays a list of all customers showing the customer's first name,

last name, City, phone number and fax. Sort the results by customer fax number in ascending order

*/

SELECT customer_first_name, customer_last_name, customer_city, customer_phone, customer_fax

FROM customers

ORDER BY customer_fax asc;

/* 3. Write a query that displays all the customers from New York or New Jersey in the "Customers"

table.

*/

```
SELECT *
FROM customers
WHERE customer_city IN ('New York', 'New Jersey');
```

/* 4. Write a query that displays all the customers from the state of California and live in

San Francisco. */

```
SELECT *
FROM customers
WHERE customer_state = 'CA' AND customer_city IN ('San Francisco');
```

/* 5. Write a query that displays each customer name as a single field in the format “firstname lastname”

with a heading of Customer, along with their phone number with a heading of Phone. Use the IN operator to

only display customers in New York, New Jersey, or Washington D.C. Sort the results by phone number.

*/

```
SELECT CONCAT(customer_first_name, ' ' , customer_last_name) as "Customer",
customer_phone as "Phone"
FROM customers
WHERE customer_city IN ('New York', 'New Jersey', 'Washington D.C.')
ORDER BY customer_phone
```

/* 6. Write a query that will list all the cities that have customers with a heading of Cities.

Only list each city once (no duplicates) and sort in descending alphabetical order. */

```
SELECT DISTINCT customer_city as "Cities"
FROM customers
ORDER BY customer_city desc;

/* 7. Write a query that displays the title of each item along with the price
(with a heading of Original) and a calculated field reflecting the price with
a 15% discount (with a heading of Sale). Display the sale price with two
decimal places using the ROUND function. Sort byprice from highest to lowest.
*/

SELECT title, unit_price as "Original", ROUND(unit_price * 0.85, 2) as "Sale"
FROM items
ORDER BY unit_price desc;

/* 8. Write a query that displays the number of orders. */

SELECT COUNT(*) as "Number Of Orders"
FROM orders;

/* 9.	Write a query that displays the customer city, first name, last name,
and zip code from the customer's table.  Use the LIKE operator to only display
customers that reside in any zip code beginning with 9.
*/

SELECT customer_city, customer_first_name, customer_last_name, customer_zip
FROM customers
WHERE customer_zip LIKE '9%';
```

```
/* 10. Write a query that displays the order id and order date for any orders placed from March
1, 2014 through April 30, 2014. Do this WITHOUT using the BETWEEN clauses. Format
the date field as Month dd, yyyy and use a heading of “Ordered”.
*/

SELECT order_id, FORMAT(order_date, 'MMMM dd, yyyy') as "Ordered"
FROM orders
WHERE order_date >= 'March 1, 2014' AND order_date <= 'April 30, 2014';

/* 11. Write a query that displays the order id and order date for any orders placed during the
month of May 2014. Do this using the BETWEEN clauses. Format the date field as
mm/dd/yy and use a heading of “Ordered”
*/

SELECT order_id, order_date, FORMAT(order_date, 'MMMM dd, yyyy') as "Ordered"
FROM orders
WHERE order_date BETWEEN 'May 1, 2014' AND 'May 30, 2014';

/* 12.  Write a query which displays the order id, customer id, and the number of days between
the order date and the ship date (use the DATEDIFF function).  Name this column “Days” and
sort
by highest to lowest number of days.  Only display orders where this result is 15 days or more.
*/

SELECT order_id, customer_id, DATEDIFF(DAY, order_date, shipped_date) AS "Days"
FROM orders
WHERE DATEDIFF(DAY, order_date, shipped_date) >= 15
ORDER BY "Days" desc;
```

/*

13. Write a query which displays the order id, customer id, employee id, and order date for all

orders that have NOT been shipped, sorted by order date with the most recent order at the top.

*/

```
SELECT order_id, customer_id, employee_id, order_date
FROM orders
WHERE shipped_date IS NULL
ORDER BY order_date desc;
```

/*

14. The Marketing Department has requested a new report of shipped orders for which the order

was placed on either a Saturday or a Sunday. Write a query which displays the order id, order

date, shipped date, along with a calculated column labeled “Order_Day” showing the day of the

week the order was placed (use the DAYNAME function). Only display orders that have shipped

and were placed on a Saturday or Sunday. Sort by order date with most recent orders at the top.

*/

```
SELECT order_id, order_date, shipped_date, DATENAME(WEEKDAY, order_date) AS 'Order_Day'
FROM orders
WHERE shipped_date IS NOT NULL and DATENAME(WEEKDAY, order_date) IN ('Saturday', 'Sunday')
ORDER BY order_date desc;
```

```
/*
15.	Write a query to display the customer's last name, phone number, and fax number but only
display those customers that have a fax number.
*/

SELECT customer_last_name, customer_phone, customer_fax
FROM customers
WHERE customer_fax IS NOT NULL;

/*
16.	For each Item, retrieve the item id, title of the item, name of the artist, price of the item.
(use JOIN operation)
*/

SELECT items.item_id, items.title, artists.artist_name, items.unit_price
FROM items
JOIN artists  ON items.artist_id=artists.artist_id;

/*
17.	Write a query that displays the customer id, customer name, order id, and employee id. Sort the
results by customer id, order id, employee id. Use LEFT JOIN operator.
*/

SELECT customers.customer_id, customers.customer_first_name,
customers.customer_last_name, orders.order_id, orders.employee_id
FROM customers
LEFT JOIN orders ON customers.customer_id=orders.customer_id
```

```
ORDER BY customers.customer_id, orders.order_id, orders.employee_id;

/*
18. List customer identification number, customer name, order number and order date for all orders
listed in the order table. Include the order number, even if there is no customer name, and
identification number available. (RIGHT OUTER JOIN)
*/

SELECT customers.customer_id, customers.customer_first_name,
customers.customer_last_name, orders.order_id, orders.order_date
FROM customers
RIGHT OUTER JOIN orders ON customers.customer_id=orders.customer_id;

/*
19. Write the name and address of the customer who placed order number 45.
*/

SELECT customer_first_name, customer_last_name, customer_address
FROM customers
JOIN orders ON customers.customer_id=orders.customer_id
WHERE order_id = '45';

/*
20. List the details about the item with the highest standard price.
*/

SELECT item_id, title, artist_id, unit_price
FROM items
WHERE unit_price =
```

```
(SELECT MAX(unit_price)
FROM items);
```

/* 21. Create a statement to insert a new record into the items table with the following values:

item_id: 11

title: Eugenio Gattinara

Artist_id: 17

unit_price: 23.51

Show your INSERT statement along with the results of the following SELECT query to verify that

the insert worked correctly.

select * from items where item_id > 9;

*/

```
INSERT INTO items (item_id, title, artist_id, unit_price)
VALUES ('11', 'Eugenio Gattinara', '17', '23.51');
SELECT *
FROM items
WHERE item_id > 9;
```

/* 22. Create a statement to update the record inserted in the previous step to change the unit price

of this item to $19.05.

item_id: 11

title: Eugenio Gattinara

artist: 17

unit_price: 19.05

Show your UPDATE statement along with the results of the following SELECT query to verify that the

insert worked correctly.

select * from items where item_id > 9;

*/

```
UPDATE items
SET unit_price = 19.05
WHERE item_id = 11;
SELECT *
FROM items
WHERE item_id > 9;
```

/* 23. Create a statement to delete the entire record that was inserted and then updated in the previous

steps. Show your DELETE statement along with the results of the following SELECT query to verify that

the insert worked correctly.

select * from items where item_id > 9;

*/

```
DELETE
FROM items
WHERE item_id = 11;
SELECT *
FROM items
WHERE item_id > 9;
```

/* 24. Using the SUBSTRING and CONCAT functions, write a query to display each customer name as a single

field in the format “Jones, Tom” with a heading of Customer along with the customer_phone field in a

nicely formatted calculated column named Phone. For example, a record containing the customer_phone

value 9095595443 would be output with parentheses, spaces, and hyphens, like this: (909) 559-5443.

Sort by first name.

*/

```
SELECT
CONCAT(customer_last_name, ', ', customer_first_name) AS "Customer",
CONCAT('(', SUBSTRING(customer_phone, 1, 3), ') ', SUBSTRING(customer_phone, 4, 3), '-', SUBSTRING(customer_phone, 7, 4)) AS "Phone"
FROM customers
ORDER BY customer_first_name;
```

/* 25. Create a statement to insert a new record with your values: your customer id, first name, last name,

address, city, state, zip code and fax number.

*/

```
INSERT INTO customers (customer_id, customer_first_name, customer_last_name, customer_address, customer_city, customer_state, customer_zip, customer_phone, customer_fax)
VALUES ('27', 'Safa', 'Alasady', '3801 W Temple Ave', 'Pomona', 'CA', '91768', '9096850815', '9096850815');
SELECT *
FROM customers
WHERE customer_id = 27;
```

/* 26. Creates a view that selects every title in the "item" table with a price higher than the average price

```
*/

USE Alasady;
GO
CREATE VIEW price_higher_than_average AS
SELECT title
FROM items
WHERE unit_price > (SELECT AVG(unit_price) FROM items);
GO
SELECT *
FROM price_higher_than_average;


/* 27. Explain the cardinality from the employees-to-employees table.
The cardinality from the employees-to-employees table is
*/

/* 28. Insert a new artist (artist id and artist name)
*/

INSERT INTO artists (artist_id, artist_name)
VALUES ('17', 'artist name');
SELECT *
FROM artists;
```