2020

T-Shirt Printing Company

DATABASE DESIGN PROJECT

Topic: A printing company (or a T-Shirt printing company)

Group member	
Name	Account
Alex Jonathan Mvami Njeunje	AJMN100 (Primary: Data Created and Stored here)
Sakala Lakshmi Venkata Maurya	LVMS100

Table of Contents

A. Describe the enterprise

1. Introduction

This project deals about the t-shit printing company which contains information of employees and customers. It keeps track of orders given by customers and employee creates the printing profile of orders from the T-Shirts available in the inventory. We also have the materials that we use to print the T-Shirts. We also deals with the billing stuff like T-shirt cost to print and total order cost to print the t-shirts. Employee will be the in charge to keep track of the printing jobs.

2. System Main Functionality

The system will perform the following and more:

- Show all the pending, done, and cancelled orders from a customer.
- Show all the pending orders with related printing jobs, ordered by their estimated delivery date.
- Show all the overdue orders.
- Show all the available T-shirts.
- Show all the printing profiles.
- Create a new order
- Create a new printing profile
- Update a printing profile price
- Delete all canceled orders
- Delete all incomplete orders

3. End Users

In our project Employees and Customers acts as end users.

4. Data Obsolescence

In our project we are handling data obsolescence by deleting the unwanted records in the tables.

5. Project Idea

We got the suggestion from the professor.

B. Entity Relationship Design

1. Entity Listing and Description

#	Entity Name	Attributes	Entity Description
1	Customers	▪ cusEmail: (Unique) customer email. Used by: Customer, Employee ▪ cusName: customer name. Used by: Customer, Employee ▪ cusPhone#: customer phone number. Used by: Customer, Employee	This table gives information of customers
2	Employees	▪ empEmail: (Unique) employee email. Used by: Customer, Employee ▪ empName: emp name. Used by: Customer, Employee ▪ empPhone#: emp phone number. Used by: Customer, Employee	This table gives information of employees
3	PrintingProfiles	▪ proName: (Unique) . Used by: Customer, Employee ▪ proMode: {B&W, Colored}. Used by: Customer, Employee ▪ proSize: {Small, Medium, Large}. Used by: Customer, Employee ▪ proPosition: {Front, Back, Sleeves}. Used by: Customer, Employee ▪ proDescription: . Used by: Customer, Employee ▪ proEstTime: Estimated time needed for one print of the printing profile. Used by: Customer, Employee ▪ proPrice: Employee estimated price for the printing profile. Used by: Customer, Employee	This table gives information of printing profiles created by employees

4	PrintingJobs	▪ jobQuantity: Number of prints for this job. . Used by: Customer, Employee ▪ jobUnitPrice: (derived). {= tshPrice + proPrice}.. Used by: Customer, Employee ▪ jobTotalCost: (derived). {= jobUnitCost*jobQuantity}.. Used by: Customer, Employee ▪ jobEstTime: (Derived). {= proEstTime*jobQuantity}. Used by: Customer, Employee ▪ jobInstructions: Additional instruction from customer. . Used by: Customer, Employee	This table gives information of printing job of printing profiles. Is a weak entity
5	ArtWorks	▪ artName: (Unique) . Used by: Customer, Employee ▪ artImage: Name of image file provided by the customer. Used by: Customer, Employee	This table gives information of artwork of T-Shirts
6	PrintingMaterials	▪ matProduct#: (unique) . Used by: Customer, Employee. Used by: Customer, Employee ▪ matName: [e.x. Red ink…] ▪ matAmount: (Dynamic) . Used by: Customer, Employee	This table gives information of materials used for printing
7	TShirts	▪ tshBrand: . Used by: Customer, Employee ▪ tshName: . Used by: Customer, Employee ▪ tshColor: . Used by: Customer, Employee ▪ tshSize: {x-small, small, medium, large, x-large}. Used by: Customer, Employee	This table gives information of T-Shirts available

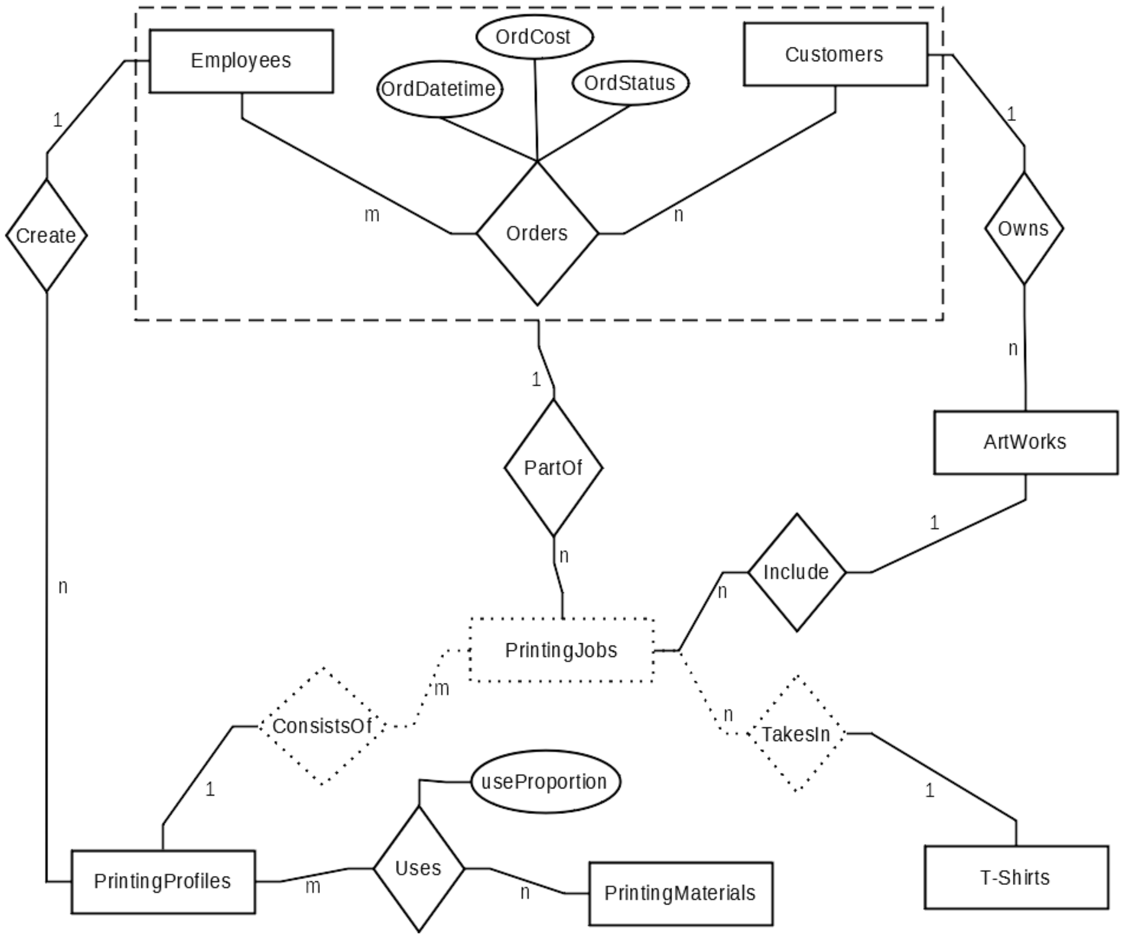
		▪ tshAmount: . Used by: Customer, Employee ▪ tshPrice: . Used by: Customer, Employee	

2. Relationships

#	Relationship	Entities Related	Description
1	Orders	Customer to Employee	• A Customer make Orders to many Employees. • An Employee can take Orders from many Customers.
2	PartOf	Printing Job to Order	• A Printing Job can be part of a single order. • An Order can be can have many Printing Jobs.
3	Create	Employee to Printing Profile	• An Employee Can Create many Printing Profiles. • A Printing Profile can be Created by a Single Employee.
4	Owns	Customer to Artwork	• A Customer Owns many Artworks. • An Artwork can be owned by a single Customer.
5	Include	Artwork to Printing Job	• An Artwork can be included in many Printing Jobs. • A Printing Job can include a Single Artwork.
6	TakesIn	T-Shirt to Printing Job	• A T-Shirt is taken for many Printing Jobs. • A Printing Job takes in a T-Shirt.
7	ConsistsOf	Printing Profile to Printing Job	• A Printing Profile constitutes many Printing Jobs. • A Printing Job consists of a single Printing Profile.
8	Uses	Printing Material to Printing Profile	• A Printing Material is used for many Printing Profiles. • A Printing Profile uses many Printing Material.

3. E-R Diagram

The resulting ER diagram:



C. Conceptual Level

- **Primary Keys in bold**
- Foreign Keys in underlined

#	Table Name	Properties/Columns	Functional Dependencies	Comments
1	Customers	▪ **cusEmail** ▪ cusName ▪ cusPhone#	cusEmail → cusName, cusPhone#	
2	Employees	▪ **empEmail** ▪ empName ▪ empPhone#	empEmail → empName, empPhone#	
3	PrintingProfiles	▪ **proName** ▪ proMode ▪ proSize ▪ proPosition ▪ proDescription ▪ proEstTime ▪ proPrice ▪ empEmail	proName → proMode, proSize, proPosition, proDescriotion, proEstTime, proPrice	
4	PrintingJobs	▪ jobQuantity ▪ jobUnitPrice ▪ jobTotalCost ▪ jobInstructions ▪ jobEstTime ▪ **proName** ▪ **tshBrand** ▪ **tshName** ▪ **tshColor** ▪ **tshSize** ▪ **ordID** ▪ artName	proName, tshBrand, tshName, tshColor, tshSize, ordID → jobQuantity, jobUnitPrice, jobTotalCost, jobInstructions, jobEstTime	
5	Artworks	▪ **artName** ▪ artImage ▪ cusEmail	artName → artImage	
6	PrintingMaterials	▪ **matProduct#** ▪ matName ▪ matAmount	matProduct# → matName, matAmount	

7	Tshirts	▪ **tshBrand** ▪ **tshName** ▪ **tshColor** ▪ **tshSize** ▪ tshAmount ▪ tshPrice	tshBrand, tshName, tshColor, tshSize → tshAmount, tshPrice	
8	Orders	▪ **ordID**: (auto generated starting with 1) ▪ ordDatetime ▪ ordCost ▪ ordStatus: {Pending, Done, Delivered, Cancelled} ▪ ordEstDeliveryDate ▪ <u>empEmail</u> ▪ <u>cusEmail</u>	ordID → ordDateTime, ordCost, ordStatus, ordEstDeliveryDate	This table gives information of orders given by customers
9	Use	▪ useProportion ▪ <u>**proName**</u> ▪ <u>**matProduct#**</u>	proName, matProduct# → useProportion	This table gives information of proportion of materials used in the printing profile

D. External View

USERS / TABLES/VIEWS	Employee	Customer
CUSTOMERS	SELECT, INSERT, UPDATE	SELECT, INSERT, UPDATE
EMPLOYEES	SELECT, INSERT, UPDATE	NONE
PRINTINGPROFILES	INSERT, UPDATE	NONE
PRINTINGJOBS	INSERT, UPDATE	INSERT, UPDATE
ARTWORKS	SELECT	SELECT, INSERT, UPDATE
PRINTINGMATERIALS	SELECT, INSERT, UPDATE	NONE
TSHIRTS	INSERT, UPDATE	NONE
ORDERS	INSERT, UPDATE	INSERT, UPDATE
USES	INSERT, UPDATE	NONE
CUSTOMERORDERS	SELECT	SELECT
PENDINGORDERS	SELECT	NONE
PROFILESLIST	SELECT	SELECT
CUSTOMERPRODUCTHISTORY	NONE	SELECT
AVAILABLETSHIRTS	SELECT	SELECT
OVERDUEORDERS	SELECT	NONE

[Create the users in the database]

E. Internal View

1. Frequent Queries

Each table Identifies a most frequent query, it's optimization and file structure needed.

Query 1	**What is the list of all pending, done, and caneled orders passed by a customer given the customer email?**
User	Customer, Employee

SQL	
	```
	CREATE VIEW customerorders
	AS
	   (SELECT orders.orddatetime,
	           orders.ordestdeliverydate,
	           orders.ordstatus,
	           orders.ordtotalcost,
	           printingjobs.proname,
	           printingjobs.tshbrand,
	           printingjobs.tshname,
	           printingjobs.tshcolor,
	```

```
               printingjobs.tshsize,
               printingjobs.artname,
               printingjobs.jobinstructions,
               orders.cusemail
     FROM      orders
               INNER JOIN printingjobs
                       ON ( orders.ordid = printingjobs.ordid )
);
```

Relational Algebra	$\pi_{(\text{orders.orddatetime, orders.ordestdeliverydate, orders.ordstatus, orders.ordtotalcost, printingjobs.proname, printingjobs.tshbrand, printingjobs.tshname, printingjobs.tshcolor, printingjobs.tshsize, printingjobs.artname, printingjobs.jobinstructions, orders.cusemail})}$ [orders $\bowtie_{[\text{ orders.ordid = printingjobs.ordid }]}$ printingjobs]
Optimization	Already optimal!
Candidate File Structures	▪ Build a Cluster File on ordID in PrintingJobs ▪ Build a Cluster File on ordID in Orders

Query 2	**What are all the poending orders with related printing jobs, ordered by the estimated delivery date.**
User	Employee

SQL

```
CREATE VIEW pendingorders
AS
   (SELECT orders.ordid,
           orders.ordestdeliverydate,
           orders.ordtotalcost,
           printingjobs.jobquantity,
           printingjobs.jobinstructions,
           printingjobs.proname,
           printingjobs.tshbrand,
           printingjobs.tshname,
           printingjobs.tshsize,
           printingjobs.tshcolor
    FROM   orders
           INNER JOIN printingjobs
                   ON ( orders.ordid = printingjobs.ordid )
```

	`    WHERE  orders.ordstatus LIKE 'pending');`
Relational Algebra	π(orders.ordid, orders.ordestdeliverydate, orders.ordtotalcost, printingjobs.jobquantity, printingjobs.jobinstructions, printingjobs.proname, printingjobs.tshbrand, printingjobs.tshname, printingjobs.tshsize, printingjobs.tshcolor) $\sigma_{(orders.ordstatus = 'pending')}$ [orders $\bowtie_{[orders.ordid = printingjobs.ordid]}$ printingjobs]
Optimization	π(orders.ordid, orders.ordestdeliverydate, orders.ordtotalcost, printingjobs.jobquantity, printingjobs.jobinstructions, printingjobs.proname, printingjobs.tshbrand, printingjobs.tshname, printingjobs.tshsize, printingjobs.tshcolor) [$\sigma_{(orders.ordstatus = 'pending')}$orders $\bowtie_{[orders.ordid = printingjobs.ordid]}$ printingjobs]
Candidate File Structures	▪ Build a Secondary B-Tree on ordStatus in Orders

Query 3	**Provide the list of all overdue orders.**
User	**Employee**

SQL	`CREATE VIEW overdueorders`
	`AS`
	`  (SELECT empemail,`
	`          orddatetime,`
	`          ordestdeliverydate,`
	`          ordtotalcost`
	`   FROM   orders`
	`   WHERE  ordestdeliverydate < sysdate);`

Relational Algebra	π(empemail, orddatetime, ordestdeliverydate, ordtotalcost) [$\sigma_{(ordestdeliverydate < sysdate)}$orders]
Optimization	Already optimal!

Candidate File Structures	✓ Build a Cluster B-Tree in Orders

Query 4	**Provide the list of all the available t-shirts.**
User	**Customer, Employee**

SQL	`CREATE VIEW availabletshirts`
	`AS`
	`  (SELECT tshbrand,`
	`          tshname,`
	`          tshcolor,`
	`          tshsize,`
	`          tshamount,`
	`          tshprice`
	`   FROM   tshirts`
	`   WHERE  tshamount <> 0);`

Relational Algebra	$\pi_{(tshbrand, tshname, tshcolor, tshsize, tshamount, tshprice)}$ [$\sigma_{(tshamount <> 0)}$tshirts]
Optimization	Already Optimal!
Candidate File Structures	▪ Build a Cluster B-Tree in Tshirts

Query 5	**Provide the list of all profiles.**
User	**Customer, Employee**

SQL	`CREATE VIEW profileslist`
	`AS`
	`  (SELECT proname,`
	`          promode,`
	`          prosize,`
	`          prodescription,`
	`          proprice`
	`   FROM   printingprofiles);`

Relational Algebra	$\pi_{(proname, promode, prosize, prodescription, proprice)}$ [$\sigma_{(tshamount <> 0)}$printingprofiles]
Optimization	Already optimal!

Candidate File Structures	▪ Leave default Secondary B-Tree indexing

Query 6	**What are the printing profiles and T-Shirts ever used by a Customers given the Customers email?**
User	**Customer**

SQL	`CREATE VIEW customerproducthistory`
	`AS`
	`  (SELECT orders.cusemail,`
	`          printingprofiles.proname,`
	`          printingprofiles.proprice,`
	`          tshirts.tshbrand,`
	`          tshirts.tshname,`
	`          tshirts.tshsize,`
	`          tshirts.tshcolor,`
	`          tshirts.tshprice`
	`   FROM   printingjobs`
	`          INNER JOIN printingprofiles`
	`                  ON printingjobs.proname = printingprofiles.proname`
	`          INNER JOIN tshirts`
	`                  ON ( printingjobs.tshbrand = tshirts.tshbrand`
	`                       AND printingjobs.tshname = tshirts.tshname`
	`                       AND printingjobs.tshcolor = tshirts.tshcolor`
	`                       AND printingjobs.tshsize = tshirts.tshsize )`
	`          INNER JOIN orders`
	`                  ON ( orders.ordid = printingjobs.ordid ))`

Relational Algebra	$\pi_{(orders.cusemail,}$ printingprofiles.proname, printingprofiles.proprice, tshirts.tshbrand, tshirts.tshname, tshirts.tshsize, tshirts.tshcolor, tshirts.tshprice) [[[printingjobs $\bowtie_{[\text{ printingjobs.proname = printingprofiles.proname}]}$ printingprofiles] $\bowtie_{[\text{ printingjobs.tshbrand = tshirts.tshbrand}}$ $\wedge$ printingjobs.tshname = tshirts.tshname $\wedge$ printingjobs.tshcolor = tshirts.tshcolor $\wedge$ printingjobs.tshsize = tshirts.tshsize] printingprofiles] $\bowtie_{[\text{ orders.ordid = printingjobs.ordid}]}$ orders]

Optimization	Already optimal!
Candidate File Structures	▪ Build a Cluster File on proName in PrintingJobs ▪ Build a Cluster File on proName in PrintingProfiles

a. File structure implemented

Synopsis of the actual file structures that will be implemented:

- ✓ Build a Cluster B-Tree in Orders
- ✓ Build a Cluster B-Tree in Tshirts
- ✓ Build a Cluster File on proName in PrintingJobs
- ✓ Build a Cluster File on proName in PrintingProfiles
- ✓ Build a Secondary B-Tree on ordStatus in Orders

2. More commands

The following is a list of some other implemented commands:

1. Update: Update a printing profile.
2. Delete: Delete all canceled orders
3. Insert: Create a new order
4. Insert: Create a new printing profile
5. Insert: Create a new art work
6. Data Obsolescence command: Delete all incomplete orders.

F. Data Dictionary

#	Name	Type	Definition
1	CUSTOMERS	Table	This table gives information of customers
2	EMPLOYEES	Table	This table gives information of employees
3	PRINTINGPROFILES	Table	This table gives information of printing profiles created by employees
4	PRINTINGJOBS	Table	This table gives information of printing job of printing profiles
5	ARTWORKS	Table	This table gives information of artwork of T-Shirts
6	PRINTINGMATERIALS	Table	This table gives information of materials used for printing
7	TSHIRTS	Table	This table gives information of T-Shirts available
8	ORDERS	Table	This table gives information of orders given by customers
9	USES	Table	This table gives information of proportion of materials used in the printing profile
10	CUSTOMERORDERS	View	This view gets the list of all customer orders
11	PENDINGORDERS	View	This view gets all pending orders
12	PROFILESLIST	View	This view will get all printing profiles
13	CUSTOMERPRODUCTHISTORY	View	This view gives information about all orders of a customer
14	AVAILABLETSHIRTS	View	This view gives information about all available t-shirts
15	OVERDUEORDERS	View	This view gives information about all overdue orders
16	PRINTPRO_PRINTJOB_IDX	Index	Built this index on PRINTING JOBS table
17	ORDERS_ORDSTATUS_IDX	Index	Built this index on ORDERS table
18	SYS_IOT_TOP_315258	Index	Built this index on PRINTING MATERIALS table
19	SYS_IOT_TOP_315260	Index	Built this index on T-SHIRTS table