FALL 2015

Sports Shop Database

Final Report

COMP 2714 Database Systems

Group #12
Norman Lim
Fred Yang
David Yu
Yiao Shu

Project Supervisor:
Mr. Benjamin Yu

Table of Contents

Milestone 5

Summary

The project was a learning experience for our group and allowed us to improve our database design and SQL skills gradually over the past few months. Sports Shop Database provides an effective way for sports store managers and employees to easily process daily sales and manage the inventory and shipments.

This system supports both in-store and online purchases. Customers can find the specific item quickly without any confusion as different items in database go under their own specific categories. Customers are able to use the store locator to find the store closest to their homes for convenience.

The system also allows managers to find out which items have brought big profits and what specs/sizes of items are popular with the customers, so that the company can advertise accordingly. Meanwhile, managers can also easily track the reasons why people return certain products so that that they can make correct judgements in deciding which products to discontinue in the upcoming seasons.

Based on the features that Sports Shop Database provides, store shippers are able to monitor the status when items are delivered right after transactions, thus ensuring customers can receive their products ordered as soon as possible.

We put a lot of effort on fulfilling the project over the past few months; however, there were challenges in working with the highly normalized tables, particularly in calculating profits and presenting returning items and shipping items. To overcome these challenges, we added two columns costPrice and salePrice in table Item, and brought in three bridge tables in second and third normal form as well:

- Transaction_Item - connects tables Transaction and Item to create a M:N relationship.
- Delivery_Item - connects tables Delivery and Item to create a M:N relationship.
- Return_Item - connects tables Return and Item to create a M:N relationship.

If we were to start designing the database all over again, we would:

- Do further requirements analysis at the first stage.
- Consider second/third NF normalization at the early stages (first / second stage).
- Use Github as the source control system.

Design Rationale

Our basic approach for the database project was to define the objective, identify the entities, relationships, define and implement business rules, and finally create the application. We chose Oracle 11g Express as the target database platform, and use its featured database development and management tools to facilitate the design process.

We developed twelve tables in Sports Shop Database, and the ER diagram we made has had at least five revisions from the beginning of the project. For example, to calculate the profits correctly, two columns costPrice and salePrice were added in table Item in the stage of second NF. To track delivery items, we

brought in a bridge table Delivery_Item to connect Delivery and Item in the stage of second NF. Similar revisions were also applied to present returning items in the stage of third NF.

SQL Statements

1). User Story 1 - Kenny (Store Owner) wants to know why people return the product so he knows which products to stop selling.

```
SELECT R.RETURNID,R.REASON,
R.TRANSACTIONID,R.RETURNDATE,
I.ITEMNAME,I.SPEC,I.COSTPRICE,
I.SALEPRICE,RI.QUANTITY
FROM RETURN R
JOIN RETURN_ITEM RI
ON R.RETURNID = RI.RETURNID
JOIN ITEM I
ON RI.ITEMNO = I.ITEMNO;
```

RETURNID	REASON	TRANSA...	RETUR...	ITEMNAME	SPEC	CO...	SAL...	Q...
RT0150...	Size not correct.	TP11500...	01-NO...	Columbia Hanath Mens H...	premium	15.15	23.99	3
RT0150...	Price not correct. Sale price higher...	TP11500...	01-NO...	Wilson TDJ Composite F...	junior	27.99	34.99	1
RT0150...	Product did not match description ...	TP11500...	01-NO...	Easton Stealth CX Hocke...		158.5	179.99	2

2). User Story 2 - Ben (Store Manager) wants to find out the profits in November for each item so he knows what products to advertise more. To simplify the calculation, suppose the **expense** of each item is $4.50 roughly.

```
SELECT R.ITEMNO,I.ITEMNAME,
TO_CHAR(SUM(R.QUANTITY*(I.SALEPRICE-I.COSTPRICE-4.50)),
 '$9,999.00') AS PROFIT
FROM TRANSACTION T
JOIN TRANSACTION_ITEM R
ON T.TRANSACTIONID = R.TRANSACTIONID
JOIN ITEM I
ON R.ITEMNO = I.ITEMNO
WHERE T.SUCCEED = 1
AND TO_CHAR(T.TRANSACTIONDATE,'MM') = '11'
GROUP BY R.ITEMNO, I.ITEMNAME
ORDER BY PROFIT DESC;
```

ITEMNO	ITEMNAME	PROFIT
IT150019	Asics Gel Rocket Womens Indoor Shoes	$47.97
IT150020	Asics Gel Rocket Womens Indoor Shoes	$41.98
IT150017	Mizuno Wave Mens Indoor Shoes	$35.00
IT150016	Mizuno Wave Mens Indoor Shoes	$31.98
IT150006	Wilson NFL Official Game Football	$29.98
IT150010	Wilson NFL Competition Football	$22.50
IT150005	Reebok Realflex Womens Boots	$20.08
IT150002	Giro Bevel White Helmet	$17.16
IT150015	Easton Stealth CX Hockey Gloves	$16.99
IT150003	Nordica Nadvo Binding	$15.99
IT150007	Wilson NFL Official Football	$14.99
IT150009	Wilson LFL Ultimat Football	$9.60
IT150001	Columbia Hanath Mens Half-zip Top	$8.68
IT150008	Wilson TDJ Composite Football	$5.00
IT150004	Bauer Supreme Stick	$1.50

3). User Story 3 - Jaimie (Employee) wants customer's emails to alert them that their item has arrived in store. (Customer IDs: CU150001, CU150005, CU150010, CU150011, CU150020)

```
SELECT CUSTOMERID,
FIRSTNAME,
LASTNAME,EMAIL
FROM CUSTOMER
WHERE CUSTOMERID IN
('CU150001','CU150005',
'CU150010','CU150011','CU150020');
```

CUSTOMERID	FIRSTNAME	LASTNAME	EMAIL
CU150001	Homer	Simpson	hsimpson@gmail.com
CU150005	Ken	Dope	kdope@gmail.com
CU150010	Job	Allen	jallen@gmail.com
CU150011	Pijon	White	pwhite@gmail.com
CU150020	Kenny	Martin	kmartin@gmail.com

4). User Story 4 - Jermaine (Employee) wants to be able to check if customers have received their delivered products so the customer can be satisfied. (deliveryId: DP01500003)

```
SELECT D.DELIVERYID,D.DELIVERYDATE,
D.RECEIVEDATE,D.CONSIGNEE,
(SELECT FIRSTNAME ||' '|| LASTNAME
FROM CUSTOMER WHERE CUSTOMERID = D.CONSIGNEE)
AS CONSIGNEENAME,
I.ITEMNAME,I.SPEC,I.COSTPRICE,
```

```
I.SALEPRICE,DI.QUANTITY
 FROM DELIVERY D
JOIN DELIVERY_ITEM DI
ON D.DELIVERYID = DI.DELIVERYID
JOIN ITEM I
ON DI.ITEMNO = I.ITEMNO
WHERE D.DELIVERYID IN ('DP01500003');
```

DELIVERYID	DELIVERY...	RECEIVEDATE	CO...	CONSIGNEENAME	ITEMNAME	SPEC	C...	S..	QU...
DP01500003	01-NOV-15	05-NOV-15	CU1...	Alex Clark	Bauer Vapor X40 Hock...	senior	58.5	6..	3

5). User Story 5 - John (administrator) wants the item to be delivered right after the transaction is made so customers can receive their product as soon as possible. (transactionId: TP1500015)

```
SELECT * FROM DELIVERY
WHERE TRANSACTIONID IN ('TP11500015');
```

DELIVERYID	TRANSACTIONID	SHIPPER	CONSIGNEE	DELIVERYDATE	RECEIVEDATE
DP01500006	TP11500015	00018	CU150012	01-NOV-15	

6). User Story 6 - Bob (Employee) wants all the football and hockey items to go under their own specific shelf so customers can find the items more easily and not get confused when looking for specific equipment.

```
SELECT C.CATNAME,I.ITEMNAME,
I.SPEC,I.SALEPRICE,I.QUANTITY
FROM ITEM I
JOIN CATEGORY C
ON I.CATNO = C.CATNO
WHERE C.CATNAME LIKE 'Equipment%'
OR C.CATNAME LIKE 'Hockey%';
```

CATNAME	ITEMNAME	SPEC	SALEPRICE	QUANTITY
Equipment	Nordica Nadvo Binding	junior	220.99	25
Equipment	Wilson NFL Competition Football	premium	29.99	20
Equipment	Wilson LFL Ultimat Football	premium	37.99	15
Equipment	Wilson TDJ Composite Football	junior	34.99	30
Equipment	Wilson NFL Official Football	senior	109.99	20
Equipment	Wilson NFL Official Game Football	senior	119.99	20
Hockey	CCM Ultra Tacks Shoulder Pads		189.99	10
Hockey	Easton Stealth CX Hockey Gloves		179.99	20
Hockey	Bauer Supreme Stick	flex75	43.99	10
Hockey	Easton V9E Hockey Stick	flex85	119.99	20
Hockey	Reebok Ribcor SC87 Hockey Skates	senior	129.99	10
Hockey	Bauer Vapor X40 Hockey Skates	senior	69.99	20

7). User Story 7 - James (employee) wants to track the batch deliveries (at least 3 items ordered at a time).

```
SELECT D.DELIVERYID,D.DELIVERYDATE,
D.RECEIVEDATE,D.CONSIGNEE,
(SELECT FIRSTNAME ||' '|| LASTNAME
FROM CUSTOMER WHERE CUSTOMERID = D.CONSIGNEE)
AS CONSIGNEENAME,
I.ITEMNAME,I.SPEC,I.COSTPRICE,
I.SALEPRICE,DI.QUANTITY
 FROM DELIVERY D
JOIN DELIVERY_ITEM DI
ON D.DELIVERYID = DI.DELIVERYID
JOIN ITEM I
ON DI.ITEMNO = I.ITEMNO
WHERE DI.QUANTITY >= 3;
```

DELIVERYID	DELIVE...	RECEIVED...	CO...	CONSIGNEENAME	ITEMNAME	SPEC	C...	S..	QUANTITY
DP01500001	01-NOV...	04-NOV-15	CU...	Mara Harper	Bauer Supreme Stick	flex75	3...	4..	3
DP01500003	01-NOV...	05-NOV-15	CU...	Alex Clark	Bauer Vapor X40 Ho...	senior	58.5	6..	3

8). User Story 8 – After purchases, Jamie (customer) wants to check points in his account. (customerId: CU150001)

```
SELECT CUSTOMERID,POINTS FROM CUSTOMER
WHERE CUSTOMERID IN ('CU150001');
```

CUSTOMERID	POINTS
CU150001	100

9). User Story 9 - Jerry (customer) wants to make many purchases at the sports store, so for convenience, he always uses the store located closest to his house.

```
SELECT * FROM STORE WHERE ADDRESS LIKE '%Vancouver%';
```

STORENO	ADDRESS	PHONE
SC-03	18 West Broadway, Vancouver, BC V5Y 1P2	604464...
SC-04	8125 Ontario Street, Vancouver, BC V5X 0A7	604464...

10). User Story 10 - Harold (Employee) wants customers to purchase their items in a single transaction so no confusion or disorganization occurs.

```
SELECT * FROM TRANSACTION;
```

TRANSACTIONID	RECEIPTNO	PAYMENTNO	CUSTOMERID	CASHIERNO	TRANSACTIONDATE	SUCCEED
TP11500001	SC11512345	VISA-7289011123	CU150001	00009	01-NOV-15	1
TP11500002	SC11512346	MAST-1179011124		00010	01-NOV-15	1
TP11500003	SC11512347	MAST-3459011123	CU150003	00011	01-NOV-15	1
TP11500004	SC11512348	VISA-7287011181	CU150004		01-NOV-15	1
TP11500005	SC11512349	VISA-3389011177	CU150005		01-NOV-15	1
TP11500006	SC11512350	VISA-7389014121	CU150006	00012	01-NOV-15	1
TP11500007	SC11512351	MAST-1279011151		00012	01-NOV-15	1
TP11500008	SC11512352	MAST-3154011123	CU150007	00013	01-NOV-15	1
TP11500009	SC11512353	VISA-1286011182	CU150008		01-NOV-15	1
TP11500010	SC11512354	VISA-3389013175	CU150008		01-NOV-15	1
TP11500011	SC11512355	VISA-7289014122	CU150009	00014	01-NOV-15	1
TP11500012	SC11512356	MAST-1179011125		00015	01-NOV-15	1
TP11500013	SC11512357	MAST-3415011123	CU150010	00015	01-NOV-15	1
TP11500014	SC11512358	VISA-0287011151	CU150011		01-NOV-15	1
TP11500015	SC11512359	VISA-3189011117	CU150012		01-NOV-15	1

11). User Story 11 - Hassan (Employee) always reports back to his manager (Kobe) before he leaves work so he can form a better relationship with his boss. (StoreNo: SC-01)

```
SELECT EMPLOYEENO,FIRSTNAME,
LASTNAME,PHONE,EMAIL,
POSITION,STORENO
FROM EMPLOYEE
WHERE STORENO = 'SC-01'
AND POSITION LIKE 'Store Manager';
```

EMPLOYEENO	FIRSTNAME	LASTNAME	PHONE	EMAIL	POSITION	STORENO
00001	Kim	Bell	6047771234	kbell@gmail.com	Store Manager	SC-01

12). User Story 12 - Henry (Customer) wants to always have someone there to help him purchase footwear, so he always asks Jenny (Employee) who is the regular employee that knows his foot size and what he needs. (Catname: Footwear, size: 11)

```
SELECT I.ITEMNAME,C.CATNAME,
I.SPEC,I.SALEPRICE,I.QUANTITY,
I.PRODUCER FROM ITEM I
JOIN CATEGORY C
ON I.CATNO = C.CATNO
WHERE C.CATNAME LIKE 'Footwear%'
AND I.SPEC LIKE '%#11%';
```

ITEMNAME	CATNAME	SPEC	SALEPRICE	QUANTITY	PRODUCER
Mizuno Wave Mens Indoor Shoes	Footwear	size#11	82.99	30	Mizuno

13). Find which customer made most purchases in November

```
SELECT T.CUSTOMERID,
(SELECT C.FIRSTNAME || ' ' || C.LASTNAME
  FROM CUSTOMER C WHERE C.CUSTOMERID = T.CUSTOMERID) AS NAME,
COUNT(DISTINCT T.TRANSACTIONID) AS MOSTPURCHASES
FROM TRANSACTION T
WHERE T.SUCCEED = 1
AND T.CUSTOMERID IS NOT NULL
AND TO_CHAR(T.TRANSACTIONDATE,'MM') = '11'
GROUP BY T.CUSTOMERID
HAVING COUNT(DISTINCT T.TRANSACTIONID) >= ALL
(SELECT COUNT(DISTINCT T1.TRANSACTIONID) FROM TRANSACTION T1
WHERE T1.SUCCEED = 1 AND T1.CUSTOMERID IS NOT NULL
AND TO_CHAR(T1.TRANSACTIONDATE,'MM') = '11'
GROUP BY T1.CUSTOMERID
);
```

CUSTOMERID	NAME	MOSTPURCHASES
CU150008	Alex Clark	2

14). List all deliveries that are labeled as Received

```
SELECT D.DELIVERYID,D.TRANSACTIONID,D.CONSIGNEE,
 (SELECT C.FIRSTNAME || ' ' || C.LASTNAME
  FROM CUSTOMER C WHERE C.CUSTOMERID = D.CONSIGNEE) AS NAME,
 (SELECT C.ADDRESS FROM CUSTOMER C
  WHERE C.CUSTOMERID = D.CONSIGNEE) AS ADDRESS,
  D.RECEIVEDATE
 FROM DELIVERY D
 WHERE D.RECEIVEDATE IS NOT NULL;
```

DELIVERYID	TRANSACTIONID	CONSIGNEE	NAME	ADDRESS	RECEIVEDATE
DP01500001	TP11500004	CU150004	Mara Harper	777 Johns cr., Surrey, BC V1B 2H4	04-NOV-15
DP01500002	TP11500005	CU150005	Ken Dope	332 Chequeen st., Coquitlam, BC V3...	04-NOV-15
DP01500003	TP11500009	CU150008	Alex Clark	308-411 Zoob road, Vancouver, BC V...	05-NOV-15
DP01500004	TP11500010	CU150008	Alex Clark	308-411 Zoob road, Vancouver, BC V...	06-NOV-15

Milestone 4

Normalized ER diagram

ERD

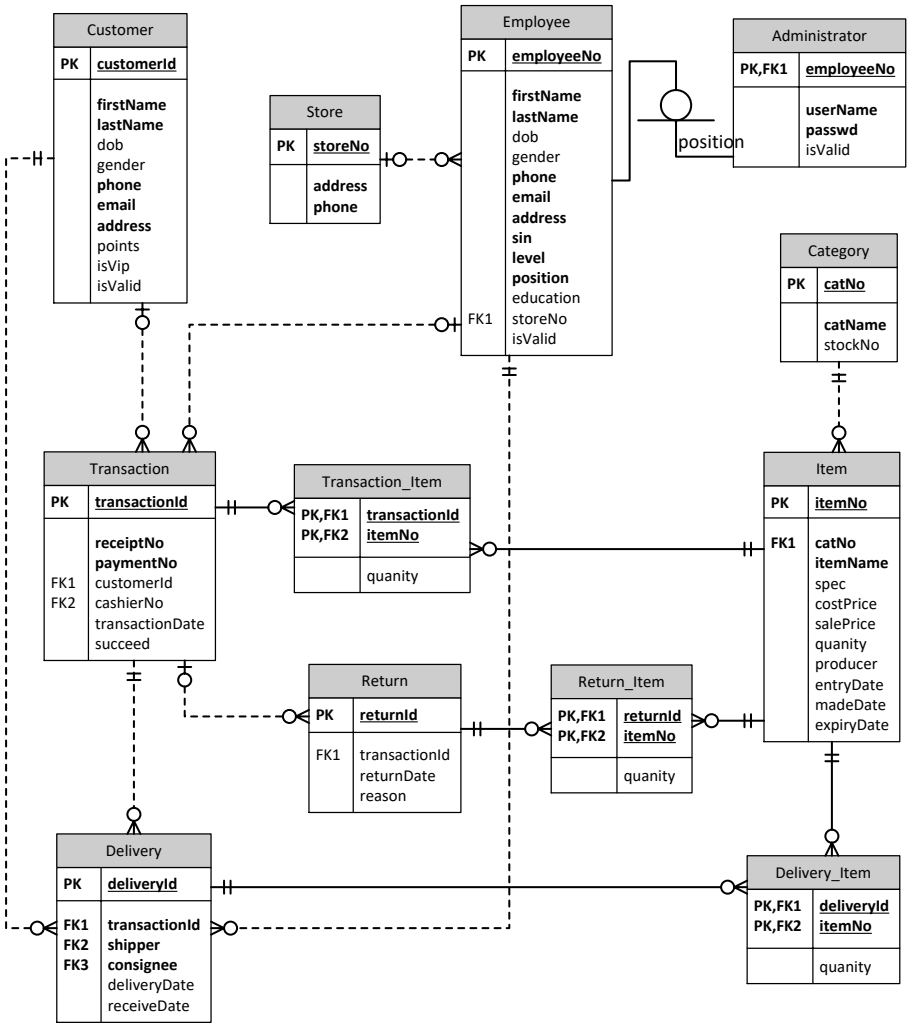


Table Normalization

Legend:

Name – Primary Key

Name – Foreign Key

Name - Attribute

Zero Normal Form (0NF)

Store (storeNo, address, phone)

Customer (customerId, firstName, lastName, dob, gender, phone, email, address, points, isVip, isValid)

Employee (employeeNo, firstName, lastName, dob, gender, phone, email, address, sin, lv, position, education, storeNo, isValid)

Category (catNo, catName, stockNo, itemNo1, itemName1, itemNo2, itemName2, itemNo3,…)

Transaction (transactionId, receiptNo, paymentNo, customerId, cashierNo, transactionDate, succeed, totalCost)

Delivery (deliveryId, transactionId, shipper**,** consignee, deliveryDate, receiveDate, totalCost)

Return (returnId, transactionId, customerId, storeId, returnDate, reason, totalCredit)

First Normal Form (1NF)

Store (storeNo, address, phone)

Customer (customerId, firstName, lastName, dob, gender, phone, email, address, points, isVip, isValid)

Employee (employeeNo, firstName, lastName, dob, gender, phone, email, address, sin, lv, position, education, **storeNo**, isValid)

Category (catNo, catName, stockNo)

Item (itemNo, **catNo**, itemName, price, quantity, producer, entryDate, madeDate, expiryDate)

Transaction (transactionId, receiptNo, paymentNo, **customerId, cashierNo**, **storeNo**, transactionDate, succeed, totalCost)

Delivery (deliveryId, **transactionId**, **shipper**, **consignee**, deliveryDate, receiveDate, totalCost)

Return (returnId, **transactionId**, returnDate, reason, totalCredit)

Second / Third Normal Form (2NF/3NF) –They are the same.

Store (storeNo, address, phone)

Customer (customerId, firstName, lastName, dob, gender, phone, email, address, points, isVip, isValid)

Employee (employeeNo, firstName, lastName, dob, gender, phone, email, address, sin, lv, position, education, **storeNo**, isValid)

Administrator (**employeeNo**, userName, passwd, isValid)

Category (catNo, catName, stockNo)

Item (itemNo, **catNo**, itemName, spec, costPrice, salePrice, quantity, producer, entryDate, madeDate, expiryDate)

Transaction (transactionId, receiptNo, paymentNo, **customerId, cashierNo**, transactionDate, succeed)

Delivery (deliveryId, **transactionId**, **shipper**, **consignee**, deliveryDate, receiveDate)

Return (returnId, **transactionId**, returnDate, reason)

Transaction_Item (**transactionId, itemNo**, quantity)

Delivery_Item (**deliveryId**, **itemNo**, quantity)

Return_Item (**returnId**, **itemNo**, quantity)

We brought in 3 bridge tables in Second/Third Normal Form:

- Transaction_Item connects tables **Transaction** and **Item** to create a M:N relationship.
- Delivery_Item connects tables **Delivery** and **Item** to create a M:N relationship.
- Return_Item connects tables **Return** and **Item** to create a M:N relationship.

Some other changes…

Meanwhile, the column 'totalCost' was eliminated from table Transaction as it's a derived attribute which can be calculated by the price and quantity of purchased items.

The column 'storeNo' was also removed from table Transaction as it can be determined by another column cashierNo, thus avoiding transitive dependency.

An additional column 'costPrice' was added in table Item, so we can calculate the profit based on the difference between salePrice and costPrice for the given item.

Referential Integrity

- All primary keys are unique and not null.
- All foreign keys must either match the primary keys of the associated entities or be null.
- Table Employee has a foreign key 'storeNo' which references to the primary key 'storeNo' of table Store. This foreign key is nullable, null for high level managers as they don't belong to specified stores.
- An administrator must be an employee. The primary key 'employeeNo' of table Administrator also acts as a foreign key, which references to the primary key 'employeeNo' of table Employee. This column cannot be null.

- Each item must fall in a specific category. The foreign key 'catNo' in table Item references to the primary key 'catNo' of table Category. This foreign key cannot be null. Any deletion of category will affect the corresponding items that fall in the given category.
- For each in-store purchase, the transaction must contain a cashierNo, which references to Employee (employeeNo). For each online purchase, the transaction must contain a customerId, which references to Customer (customerId).
- Each delivery must be associated with transactionId, shipper, and consignee, which reference to Transaction (transactionId), Employee (employeeNo), and Customer (customerId), respectively.
- Each return must be associated with a transactionId, which references to Transaction (transactionId).

Sample Tables with Actual Data

According to the Data Sources in M2 and stakeholder's requirements in M1, we added 5 category records in table Category and 20 records in Item so far.

Category

CATNO	CATNAME	STOCKNO
CA150001	Apparel	N01
CA150002	Outwear	N01
CA150003	Equipment	N02
CA150004	Hockey	N02
CA150005	Footwear	N03

Item

ITEMNO	CATNO	ITEMNAME	SPEC	COSTPRICE	SALEPRICE	QUANTITY	PRODUCER	ENTRYDATE	MADEDATE
IT150001	CA150001	Columbia Hanath Mens Half-zip Top	premium	15.15	23.99	50	Hanath	01-NOV-15	18-JAN-15
IT150002	CA150002	Giro Bevel White Helmet	senior	45.2	53.99	50	Giro	01-NOV-15	18-JAN-15
IT150003	CA150003	Nordica Nadvo Binding	junior	200.5	220.99	25	Nordica	01-NOV-15	01-FEB-15
IT150004	CA150004	Bauer Supreme Stick	flex75	38.99	43.99	10	Bauer	01-NOV-15	15-MAR-15
IT150005	CA150005	Reebok Realflex Womens Boots	size#9	91.45	105.99	8	Reebok	01-NOV-15	02-SEP-15
IT150006	CA150003	Wilson NFL Official Game Football	senior	100.5	119.99	20	Wilson	01-NOV-15	18-FEB-15
IT150007	CA150003	Wilson NFL Official Football	senior	90.5	109.99	20	Wilson	01-NOV-15	18-FEB-15
IT150008	CA150003	Wilson TDJ Composite Football	junior	27.99	34.99	30	Wilson	01-NOV-15	08-MAR-15
IT150009	CA150003	Wilson LFL Ultimat Football	premium	28.69	37.99	15	Wilson	01-NOV-15	18-FEB-15
IT150010	CA150003	Wilson NFL Competition Football	premium	20.99	29.99	20	Wilson	01-NOV-15	15-FEB-15
IT150011	CA150004	Bauer Vapor X40 Hockey Skates	senior	58.5	69.99	20	Bauer	01-NOV-15	18-FEB-15
IT150012	CA150004	Reebok Ribcor SC87 Hockey Skates	senior	100.5	129.99	10	Reebok	01-NOV-15	13-JAN-15
IT150013	CA150004	Easton V9E Hockey Stick	flex85	100.5	119.99	20	Easton	01-NOV-15	18-FEB-15
IT150014	CA150004	CCM Ultra Tacks Shoulder Pads		158.5	189.99	10	CCM	01-NOV-15	11-MAY-15
IT150015	CA150004	Easton Stealth CX Hockey Gloves		158.5	179.99	20	Easton	01-NOV-15	18-FEB-15
IT150016	CA150005	Mizuno Wave Mens Indoor Shoes	size#9	59.5	79.99	30	Mizuno	01-NOV-15	08-JAN-15

Here, different items go under their own specific shelf so customers can find the specific item they are looking for quickly without any confusion when looking for specific equipment.
(Project itemName, catNo, stockNo from the tables above).

If Ben (Store Manager) wants to find out which items bring him the most profit, so he knows what products to advertise more. (Find quantity purchased and difference between salePrice and costPrice). Moreover, he can also check what specs/sizes are popular with the customers.
We also added 5 records in table Store as follows,

Store

STORENO	ADDRESS	PHONE
SC-01	2929 Barnet Highway #1400, Coquitlam,...	6044645110
SC-02	4700 Kingsway, Burnaby, BC V5H 4M1	6044645112
SC-03	18 West Broadway, Vancouver, BC V5Y ...	6044645113
SC-04	8125 Ontario Street, Vancouver, BC V5X...	6044645114
SC-05	Unit 600 5771 Marine Way, Burnaby, BC ...	6044645115

If Jerry (customer) wishes to make many purchases at the sports store, he will always use the store located closest to his house for convenience. (Project storeNo, address from the table above).

We also have Transaction and Transaction_Item tables as follows:

Transaction

TRANSACTIONID	RECEIPTNO	PAYMENTNO	CUSTOMERID	CASHIERNO	TRANSACTIONDATE	SUCCEED
TP11500001	SC11512345	VISA-7289011123	CU150001	00009	01-NOV-15	1
TP11500002	SC11512346	MAST-1179011124		00010	01-NOV-15	1
TP11500003	SC11512347	MAST-3459011123	CU150003	00011	01-NOV-15	1
TP11500004	SC11512348	VISA-7287011181	CU150004		01-NOV-15	1
TP11500005	SC11512349	VISA-3389011177	CU150005		01-NOV-15	1
TP11500006	SC11512350	VISA-7389014121	CU150006	00012	01-NOV-15	1
TP11500007	SC11512351	MAST-1279011151		00012	01-NOV-15	1
TP11500008	SC11512352	MAST-3154011123	CU150007	00013	01-NOV-15	1
TP11500009	SC11512353	VISA-1286011182	CU150008		01-NOV-15	1
TP11500010	SC11512354	VISA-3389013175	CU150008		01-NOV-15	1
TP11500011	SC11512355	VISA-7289014122	CU150009	00014	01-NOV-15	1
TP11500012	SC11512356	MAST-1179011125		00015	01-NOV-15	1
TP11500013	SC11512357	MAST-3415011123	CU150010	00015	01-NOV-15	1
TP11500014	SC11512358	VISA-0287011151	CU150011		01-NOV-15	1
TP11500015	SC11512359	VISA-3189011117	CU150012		01-NOV-15	1
TP11500016	SC11512360	VISA-5286011123	CU150013	00015	01-NOV-15	1

Transaction_Item

TRANSACTIONID	ITEMNO	QUANTITY
TP11500001	IT150001	2
TP11500002	IT150003	1
TP11500003	IT150005	2
TP11500004	IT150004	3
TP11500005	IT150002	4
TP11500006	IT150008	2
TP11500007	IT150007	1
TP11500008	IT150006	2
TP11500009	IT150009	2
TP11500010	IT150010	5
TP11500011	IT150020	2
TP11500012	IT150019	3
TP11500013	IT150016	2
TP11500014	IT150015	1
TP11500015	IT150017	2

From the above tables, if Harold (Employee) wants customers to purchase their items in a single transaction so no confusion or disorganization occurs, he can easily check transactionId, receiptNo, customerId, items and quantity purchased from the above tables.

Then, Delivery and Delivery_Item tables:

Delivery

DELIVERYID	TRANSACTIONID	SHIPPER	CONSIGNEE	DELIVERYDATE	RECEIVEDATE
DP01500001	TP11500004	00018	CU150004	01-NOV-15	04-NOV-15
DP01500002	TP11500005	00018	CU150005	01-NOV-15	04-NOV-15
DP01500003	TP11500009	00018	CU150008	01-NOV-15	05-NOV-15
DP01500004	TP11500010	00018	CU150008	01-NOV-15	06-NOV-15
DP01500005	TP11500014	00018	CU150011	01-NOV-15	
DP01500006	TP11500015	00018	CU150012	01-NOV-15	

Delivery_Item

DELIVERYID	ITEMNO	QUANTITY
DP01500001	IT150004	3
DP01500002	IT150005	2
DP01500003	IT150011	3
DP01500004	IT150015	1
DP01500005	IT150017	1
DP01500006	IT150020	2

If John (Shipper) wants the item to be delivered right after the transaction is made so customers can receive their product as soon as possible, he can track the deliveryId, transactionId, consignee, deliveryDate, receiveDate, itemNo from the tables above.

Return

RETURNID	TRANSACTIONID	RETURNDATE	REASON
RT01500001	TP11500016	01-NOV-15	Size not correct.
RT01500002	TP11500017	01-NOV-15	Price not correct. Sale price higher than ...
RT01500003	TP11500018	01-NOV-15	Product did not match description on we...

Return_Item

RETURNID	ITEMNO	QUANTITY
RT01500001	IT150001	3
RT01500002	IT150008	1
RT01500003	IT150015	2

If Kenny (Store Owner) wants to know why people return the product so that he can discontinue certain purchases appropriately by easily checking returnId, transactionId, itemNo, and reason from the tables above.

Customer

CUSTOMERID	FIRS...	LAST...	DOB	GE...	PHONE	EMAIL	ADDRESS
CU150001	Homer	Simps...	01-SE...	1	604556...	hsimpson@gmail.com	515 Ontario Street, Va...
CU150002	Tina	Smith	02-JA...	0	778558...	tsmith@gmail.com	21 Maroal Dr., Richmo...
CU150003	Alex	Spotter	02-FE...	1	604156...	aspotter@hotmail.com	4147 Chatham st., Van...
CU150004	Mara	Harper	10-SE...	0	604776...	mharper@netmail.com	777 Johns cr., Surrey, ...
CU150005	Ken	Dope	05-OC...	1	604888...	kdope@gmail.com	332 Chequeen st., Coq...
CU150006	Dick	Johnson	20-SE...	1	778156...	djohnson@gmail.com	1515 Barcode Street, L...
CU150007	Honna	Wong	02-MA...	0	778558...	hwong@net.com	215 Capital Dr., Delta, ...
CU150008	Alex	Clark	12-JUL...	1	604888...	aclark@hotmail.com	308-411 Zoob road, V...
CU150009	Lana	Moore	10-SE...	0	604033...	lmoore@netmail.com	132 Hall st., New West...
CU150010	Job	Allen	30-NO...	1	604910...	jallen@gmail.com	111-3321 Chequeen St...
CU150011	Pijon	White	01-SE...	1	604802...	pwhite@gmail.com	120-5151 Ontario Stre...
CU150012	Jonisa	Davis	02-NO...	0	778118...	jdavis@gmail.com	2102 Mars Dr., Richmo...
CU150013	Pipo	Baker	01-JA...	1	604456...	pbaker@hotmail.com	147 Boomboo road, Va...
CU150014	Alisa	Green	12-SE...	0	604176...	agreen@netmail.com	77 Marphoon Hwy., Su...
CU150015	Jet	King	05-OC...	1	604367...	jking@gmail.com	300-133 Douglas Stree...
CU150016	Wakon	Lewis	02-SE...	1	604554...	wlewis@gmail.com	501 Peter Street, Vanc...

From this table, Jaimie (Employee) can easily get customers' emails in order to alert them that their items have arrived in store.

Employee

EMPLOYEENO	FIRSTNAME	LASTNAME	PHONE	EMAIL	LV	POSITION	STORENO
00001	Kim	Bell	6047771234	kbell@gmail.com	3	Store Manager	SC-01
00002	Anna	Johns	6047761238	ajohns@sports.com	2	Store Manager	SC-02
00003	Tony	Chueng	7787751234	tchueng@gmail.com	2	Administrator	SC-01
00004	Benedict	Singh	6057799234	bsingh@netmail.com	2	Administrator	SC-02
00005	Gordon	Broinne	6047771200	gbroinne@gmail.com	1	Administrator	SC-05
00006	Benney	Collins	6047741234	bcollins@gmail.com	3	Store Manager	SC-03
00007	Manina	James	6057761238	mjames@sports.com	2	Store Manager	SC-04
00008	Tony	Nelson	7781751234	tnelson@gmail.com	2	Store Manager	SC-05
00009	Bars	Proter	6055799234	bproter@netmail.com	1	Cashier	SC-01
00010	Gordon	Nekon	6041271200	gnekon@gmail.com	1	Cashier	SC-01
00011	Jomo	Bella	6047971234	jbella@gmail.com	1	Cashier	SC-02
00012	Anna	Koop	6047861235	akoop@sports.com	1	Cashier	SC-02
00013	Tonney	Elliott	7786651234	telliott@gmail.com	1	Cashier	SC-03
00014	Benot	Hudson	6056789234	bhudson@netmail.com	1	Cashier	SC-03
00015	Gord	Franklin	6047581200	gfranklin@gmail.com	1	Cashier	SC-04
00016	Duke	England	6047881234	dengland@gmail.com	1	Cashier	SC-04
00017	Amnea	Cobb	6044761228	acobb@sports.com	1	Cashier	SC-05

From the table above, Hassan can easily find whom he should talk /report to in case of emergency.

Administrator

EMPLOYEENO	USERNAME	PASSWD	ISVALID
00003	admin	202cb962ac59075b...	1
00004	scadmin	202cb962ac59075b...	1
00005	scadmin1	202cb962ac59075b...	1

From the table above, the store owner can easily find how many administrators there are within the system.

DB Schema

Formatted: Font: +Body (Calibri), 11 pt

```
-- Sports Shop Schema

DROP TABLE Transaction_Item;

DROP TABLE Delivery_Item;

DROP TABLE Return_Item;

DROP TABLE Item;

DROP TABLE Category;

DROP TABLE Return;

DROP TABLE Delivery;

DROP TABLE Transaction;

DROP TABLE Customer;

DROP TABLE Administrator;

DROP TABLE Employee;

DROP TABLE Store;


CREATE TABLE Store (

   storeNo     CHAR(5)     NOT NULL,

   address     VARCHAR(70) NOT NULL,

   phone       VARCHAR(10) NOT NULL,

   PRIMARY KEY (storeNo)
```

```
);

CREATE TABLE Customer (
    customerId  CHAR(8)     NOT NULL,
    firstName   VARCHAR(15) NOT NULL,
    lastName    VARCHAR(15) NOT NULL,
    dob         DATE,
    gender      CHAR(1),
    phone       VARCHAR(10) NOT NULL,
    email       VARCHAR(20) NOT NULL,
    address     VARCHAR(70) NOT NULL,
    points      INT,
    isVip       CHAR(1)     DEFAULT '0',
    isValid     CHAR(1)     DEFAULT '1',
    PRIMARY KEY (customerId)
);

CREATE TABLE Employee (
    employeeNo  CHAR(5)     NOT NULL,
    firstName   VARCHAR(15) NOT NULL,
    lastName    VARCHAR(15) NOT NULL,
    dob         DATE,
    gender      CHAR(1),
    phone       VARCHAR(10) NOT NULL,
    email       VARCHAR(20) NOT NULL,
    address     VARCHAR(70) NOT NULL,
    sin         CHAR(9)     NOT NULL,
    lv          SMALLINT    DEFAULT 1 NOT NULL,
    position    VARCHAR(15) NOT NULL,
```

```
    education   VARCHAR(30),
    storeNo     CHAR(5),
    isValid     CHAR(1)     DEFAULT '1',
    PRIMARY KEY (employeeNo),
    FOREIGN KEY (storeNo) REFERENCES Store(storeNo)
        ON DELETE CASCADE
);

CREATE TABLE Administrator (
    employeeNo  CHAR(5)     NOT NULL,
    userName    VARCHAR(10) NOT NULL,
    passwd      VARCHAR(32) NOT NULL,
    isValid     CHAR(1)     DEFAULT '1',
    PRIMARY KEY (employeeNo),
    FOREIGN KEY (employeeNo) REFERENCES Employee(employeeNo)
        ON DELETE CASCADE
);

CREATE TABLE Category (
    catNo       CHAR(8)     NOT NULL,
    catName     VARCHAR(20) NOT NULL,
    stockNo     CHAR(3),
    PRIMARY KEY (catNo)
);

CREATE TABLE Item (
    itemNo      CHAR(8)     NOT NULL,
    catNo       CHAR(8)     NOT NULL,
    itemName    VARCHAR(50) NOT NULL,
```

```
    spec        VARCHAR(20),
    costPrice   DECIMAL(6,2),
    salePrice   DECIMAL(6,2),
    quantity    INT,
    producer    VARCHAR(30),
    entryDate   DATE,
    madeDate    DATE,
    expiryDate  DATE,
    PRIMARY KEY (itemNo),
    FOREIGN KEY (catNo) REFERENCES Category(catNo)
        ON DELETE CASCADE
);

CREATE TABLE Transaction (
    transactionId   CHAR(10)    NOT NULL,
    receiptNo       CHAR(10)    NOT NULL,
    paymentNo       VARCHAR(15) NOT NULL,
    customerId      CHAR(8),
    cashierNo       CHAR(5),
    transactionDate DATE        NOT NULL,
    succeed         CHAR(1),
    PRIMARY KEY (transactionId),
    FOREIGN KEY (customerId) REFERENCES Customer(customerId)
        ON DELETE CASCADE,
    FOREIGN KEY (cashierNo) REFERENCES Employee(employeeNo)
        ON DELETE CASCADE
);

CREATE TABLE Delivery (
```

```
    deliveryId      CHAR(10)    NOT NULL,
    transactionId   CHAR(10)    NOT NULL,
    shipper         CHAR(5)     NOT NULL,
    consignee       CHAR(8)     NOT NULL,
    deliveryDate    DATE,
    receiveDate     DATE,
    PRIMARY KEY (deliveryId),
    FOREIGN KEY (transactionId) REFERENCES Transaction(transactionId)
        ON DELETE CASCADE,
    FOREIGN KEY (shipper) REFERENCES Employee(employeeNo)
        ON DELETE CASCADE,
    FOREIGN KEY (consignee) REFERENCES Customer(customerId)
        ON DELETE CASCADE
);

CREATE TABLE Return (
    returnId        CHAR(10)    NOT NULL,
    transactionId   CHAR(10)    NOT NULL,
    returnDate      DATE,
    reason          VARCHAR(70),
    PRIMARY KEY (returnId),
    FOREIGN KEY (transactionId) REFERENCES Transaction(transactionId)
        ON DELETE CASCADE
);

CREATE TABLE Transaction_Item (
    transactionId   CHAR(10)    NOT NULL,
    itemNo          CHAR(8)     NOT NULL,
    quantity        INT,
```

```
    PRIMARY KEY (transactionId, itemNo),
    FOREIGN KEY (transactionId) REFERENCES Transaction(transactionId)
        ON DELETE CASCADE,
    FOREIGN KEY (itemNo) REFERENCES Item(itemNo)
        ON DELETE CASCADE
);

CREATE TABLE Delivery_Item (
    deliveryId      CHAR(10)    NOT NULL,
    itemNo          CHAR(8)     NOT NULL,
    quantity        INT,
    PRIMARY KEY (deliveryId, itemNo),
    FOREIGN KEY (deliveryId) REFERENCES Delivery(deliveryId)
        ON DELETE CASCADE,
    FOREIGN KEY (itemNo) REFERENCES Item(itemNo)
        ON DELETE CASCADE
);

CREATE TABLE Return_Item (
    returnId        CHAR(10)    NOT NULL,
    itemNo          CHAR(8)     NOT NULL,
    quantity        INT,
    PRIMARY KEY (returnId, itemNo),
    FOREIGN KEY (returnId) REFERENCES Return(returnId)
        ON DELETE CASCADE,
    FOREIGN KEY (itemNo) REFERENCES Item(itemNo)
        ON DELETE CASCADE
);
```

Sample SQL Statements

~~a) List profits made in November per item~~

Formatted: Strikethrough

```
SELECT R.ITEMNO,I.ITEMNAME,
TO_CHAR(SUM(R.QUANTITY*(I.SALEPRICE-I.COSTPRICE)),
 '$9,999.00') AS PROFIT
FROM TRANSACTION T
JOIN TRANSACTION_ITEM R
ON T.TRANSACTIONID = R.TRANSACTIONID
JOIN ITEM I
ON R.ITEMNO = I.ITEMNO
WHERE T.SUCCEED = 1
AND T.TRANSACTIONDATE >= TO_DATE('20151101','YYYYMMDD')
AND T.TRANSACTIONDATE <= TO_DATE('20151130','YYYYMMDD')
GROUP BY R.ITEMNO, I.ITEMNAME
ORDER BY PROFIT DESC;
```

ITEMNO	ITEMNAME	PROFIT
IT150019	Asics Gel Rocket Womens Indoor Shoes	$61.47
IT150020	Asics Gel Rocket Womens Indoor Shoes	$50.98
IT150010	Wilson NFL Competition Football	$45.00
IT150017	Mizuno Wave Mens Indoor Shoes	$44.00
IT150016	Mizuno Wave Mens Indoor Shoes	$40.98
IT150006	Wilson NFL Official Game Football	$38.98
IT150002	Giro Bevel White Helmet	$35.16
IT150005	Reebok Realflex Womens Boots	$29.08
IT150015	Easton Stealth CX Hockey Gloves	$21.49
IT150003	Nordica Nadvo Binding	$20.49
IT150007	Wilson NFL Official Football	$19.49
IT150009	Wilson LFL Ultimat Football	$18.60
IT150001	Columbia Hanath Mens Half-zip Top	$17.68
IT150004	Bauer Supreme Stick	$15.00
IT150008	Wilson TDJ Composite Football	$14.00

a) List profits made in November per item. To simplify the calculation, suppose the **expense** of each item is $4.50 roughly.

```
SELECT R.ITEMNO,I.ITEMNAME,
TO_CHAR(SUM(R.QUANTITY*(I.SALEPRICE-I.COSTPRICE-4.50)),
 '$9,999.00') AS PROFIT
FROM TRANSACTION T
JOIN TRANSACTION_ITEM R
ON T.TRANSACTIONID = R.TRANSACTIONID
JOIN ITEM I
ON R.ITEMNO = I.ITEMNO
WHERE T.SUCCEED = 1
AND TO_CHAR(T.TRANSACTIONDATE,'MM') = '11'
```

```
GROUP BY R.ITEMNO, I.ITEMNAME
ORDER BY PROFIT DESC;
```

ITEMNO	ITEMNAME	PROFIT
IT150019	Asics Gel Rocket Womens Indoor Shoes	$47.97
IT150020	Asics Gel Rocket Womens Indoor Shoes	$41.98
IT150017	Mizuno Wave Mens Indoor Shoes	$35.00
IT150016	Mizuno Wave Mens Indoor Shoes	$31.98
IT150006	Wilson NFL Official Game Football	$29.98
IT150010	Wilson NFL Competition Football	$22.50
IT150005	Reebok Realflex Womens Boots	$20.08
IT150002	Giro Bevel White Helmet	$17.16
IT150015	Easton Stealth CX Hockey Gloves	$16.99
IT150003	Nordica Nadvo Binding	$15.99
IT150007	Wilson NFL Official Football	$14.99
IT150009	Wilson LFL Ultimat Football	$9.60
IT150001	Columbia Hanath Mens Half-zip Top	$8.68
IT150008	Wilson TDJ Composite Football	$5.00
IT150004	Bauer Supreme Stick	$1.50

~~b) Find which customer made most purchases in November~~

Formatted: Strikethrough

```
SELECT T.CUSTOMERID,
(SELECT C.FIRSTNAME || ' ' || C.LASTNAME
  FROM CUSTOMER C WHERE C.CUSTOMERID = T.CUSTOMERID) AS NAME,
COUNT(DISTINCT T.TRANSACTIONID) AS MOSTPURCHASES
FROM TRANSACTION T
WHERE T.SUCCEED = 1
AND T.CUSTOMERID IS NOT NULL
AND T.TRANSACTIONDATE >= TO_DATE('20151101','YYYYMMDD')
AND T.TRANSACTIONDATE <= TO_DATE('20151130','YYYYMMDD')
GROUP BY T.CUSTOMERID
ORDER BY MOSTPURCHASES DESC;
```

CUSTOMERID	NAME	MOSTPURCHASES
CU150008	Alex Clark	2
CU150003	Alex Spotter	1
CU150004	Mara Harper	1
CU150005	Ken Dope	1
CU150006	Dick Johnson	1
CU150007	Honna Wong	1
CU150001	Homer Simpson	1
CU150009	Lana Moore	1
CU150010	Job Allen	1
CU150011	Pijon White	1
CU150012	Jonisa Davis	1
CU150013	Pipo Baker	1
CU150015	Jet King	1

b). Find which customer made most purchases in November

```
SELECT T.CUSTOMERID,
(SELECT C.FIRSTNAME || ' ' || C.LASTNAME
  FROM CUSTOMER C WHERE C.CUSTOMERID = T.CUSTOMERID) AS NAME,
COUNT(DISTINCT T.TRANSACTIONID) AS MOSTPURCHASES
FROM TRANSACTION T
WHERE T.SUCCEED = 1
AND T.CUSTOMERID IS NOT NULL
AND TO CHAR(T.TRANSACTIONDATE,'MM') = '11'
GROUP BY T.CUSTOMERID
HAVING COUNT(DISTINCT T.TRANSACTIONID) >= ALL
(SELECT COUNT(DISTINCT T1.TRANSACTIONID) FROM TRANSACTION T1
WHERE T1.SUCCEED = 1 AND T1.CUSTOMERID IS NOT NULL
AND TO CHAR(T1.TRANSACTIONDATE,'MM') = '11'
GROUP BY T1.CUSTOMERID
);
```

CUSTOMERID	NAME	MOSTPURCHASES
CU150008	Alex Clark	2

c) List all deliveries that are labeled as Received

```
SELECT D.DELIVERYID,D.TRANSACTIONID,D.CONSIGNEE,
  (SELECT C.FIRSTNAME || ' ' || C.LASTNAME
   FROM CUSTOMER C WHERE C.CUSTOMERID = D.CONSIGNEE) AS NAME,
  (SELECT C.ADDRESS FROM CUSTOMER C
   WHERE C.CUSTOMERID = D.CONSIGNEE) AS ADDRESS,
   D.RECEIVEDATE
```

```
FROM DELIVERY D
WHERE D.RECEIVEDATE IS NOT NULL;
```

DELIVERYID	TRANSACTIONID	CONSIGNEE	NAME	ADDRESS	RECEIVEDATE
DP01500001	TP11500004	CU150004	Mara Harper	777 Johns cr., Surrey, BC V1B 2H4	04-NOV-15
DP01500002	TP11500005	CU150005	Ken Dope	332 Chequeen st., Coquitlam, BC V3...	04-NOV-15
DP01500003	TP11500009	CU150008	Alex Clark	308-411 Zoob road, Vancouver, BC V...	05-NOV-15
DP01500004	TP11500010	CU150008	Alex Clark	308-411 Zoob road, Vancouver, BC V...	06-NOV-15

Track Changes

a).We added 2 tables below for user story 1, ie. Kenny (Store Owner) wants to know why people return the product so he knows which products to stop selling.

Return

Column	Type	Nullable	Description
returnId	char(10)	not null	Primary key
transactionId	char(10)	not null	Foreign key, references to Transaction (transactionId)
returnDate	date		Returning date
reason	varchar(70)		Returning reasons

Return_Item

Column	Type	Is Null	Description
returnId	char(10)	not null	Primary key, foreign key, references to Return (returnId)
itemNo	char(8)	not null	Primary key, foreign key, references to Item (itemNo)
quantity	int		amount of items purchased

b).As administrator is a subtype of employee, we removed the column 'adminId' and used 'employeeNo' as primary key and foreign key. This change was also made to ERD.

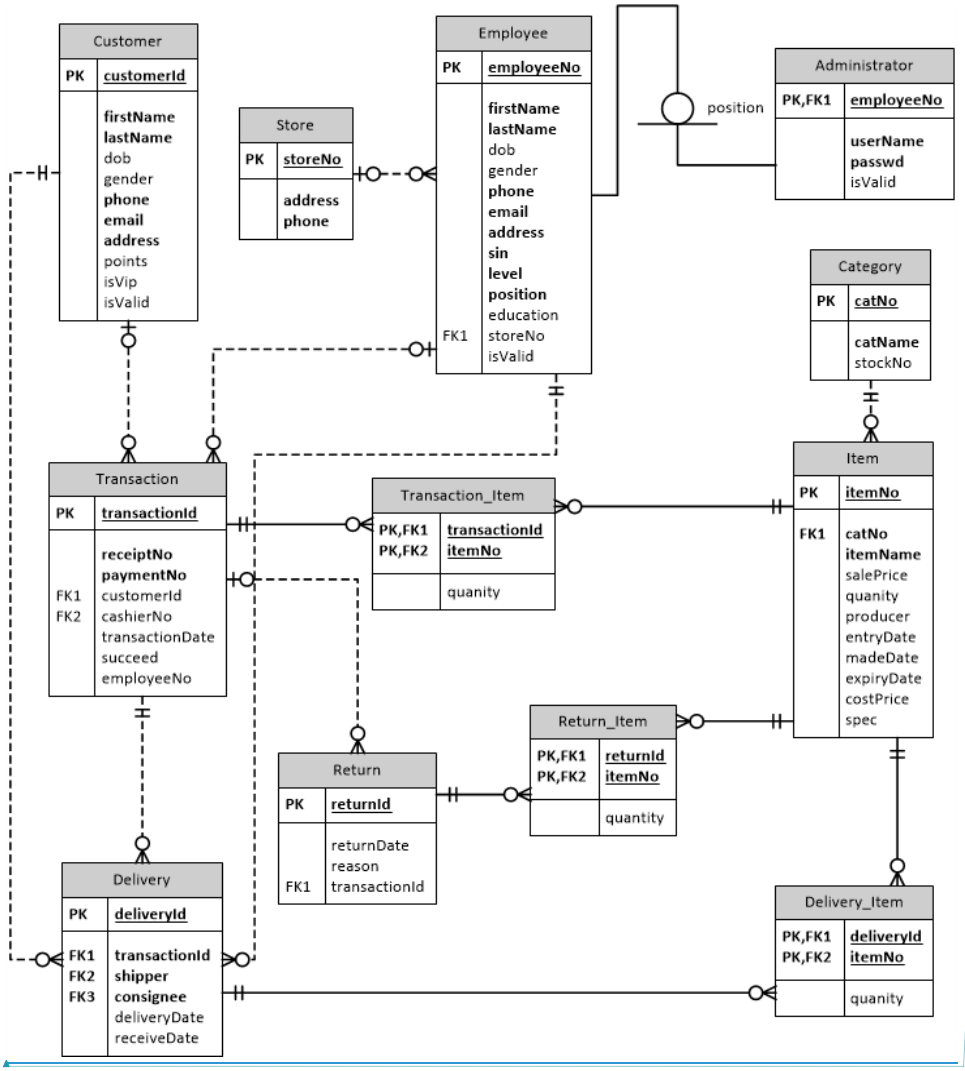
c).The column 'totalCost' was removed from table transaction, it's a derived attribute that can be calculated by quantities and sale prices of the items purchased.

d).Added 2 column 'costPrice', 'spec' to the table Item. We need both costPrice and salePrice to calculate profit of the given item.

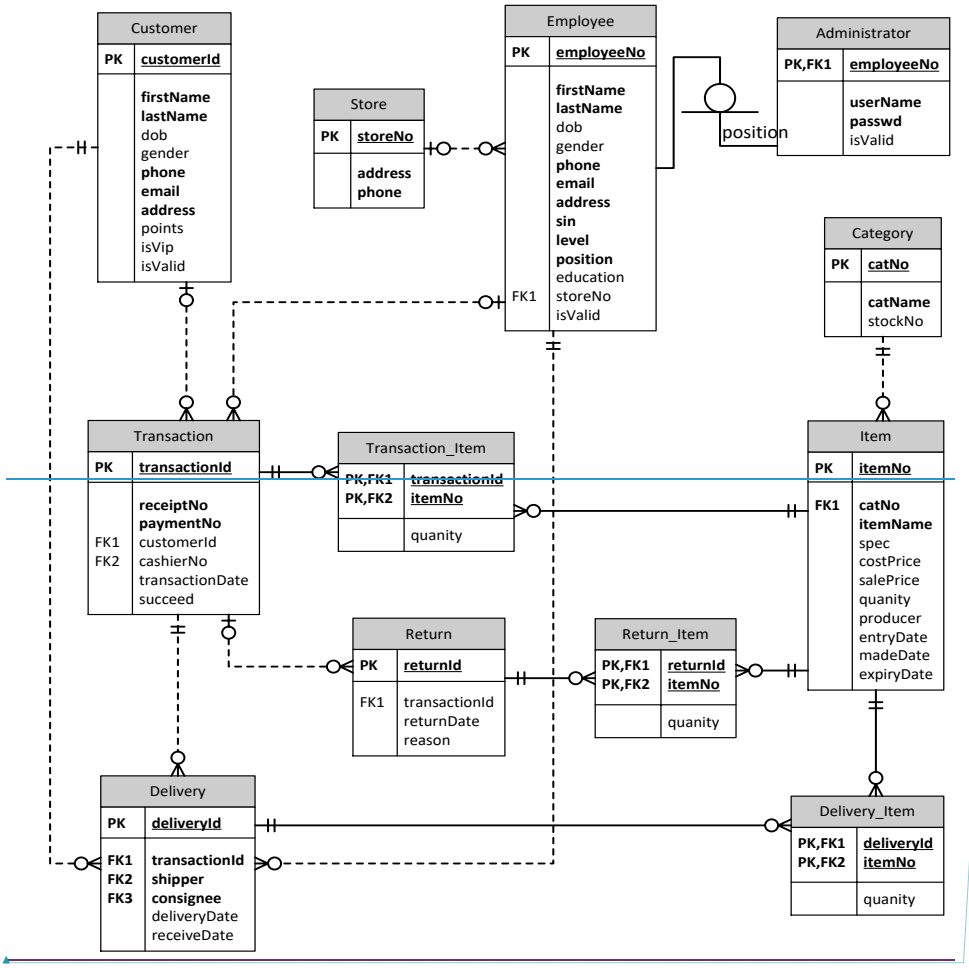
e).Added 1 column 'receiveDate' to the table Delivery. The column can be used for tracking when the customer receives the order items.

Milestone 3

ER Diagram



Formatted: Pattern: Clear (Custom Color(RGB(254,254,254)))



Field Code Changed

Design Choices

To create our ER diagram, we first had to identify the entities and their relationships. These were identified by examining the documentation in milestones 1 and 2.

Firstly, it was determined that a **CUSTOMER** can make many purchases, which generate **TRANSACTIONS**. Each **TRANSACTION** is made by a single **CUSTOMER**.

Next, a **TRANSACTION** can be performed by a single **EMPLOYEE**. Each **EMPLOYEE** can perform multiple **TRANSACTIONS**.

Each **STORE** can have multiple **EMPLOYEES** working there, and each **EMPLOYEE** can only work at a single **STORE**.

An **EMPLOYEE** may not be an **ADMINISTRATOR**, but an **ADMINISTRATOR** is an **EMPLOYEE**.

ITEMS stocked must be each be grouped into a single **CATEGORY**. Each **CATEGORY** can contain many **ITEMS**.

An **ITEM** can be a part of many **TRANSACTIONS**, and a **TRANSACTION** can contain many **ITEMS**. A bridge entity called **TRANSACTION_ITEM** was created to address this relationship.

Additionally, an **ITEM** can be a part of many **DELIVERIES**, and a **DELIVERY** can contain many **ITEMS**. A bridge entity called **DELIVERY_ITEM** was created to address this relationship.

A **DELIVERY** must be shipped by a single **EMPLOYEE**. An **EMPLOYEE** can ship many **DELIVERIES**.

A **TRANSACTION** may be required to be shipped through many **DELIVERIES**. Every **DELIVERY** must be made from a single **TRANSACTION**.

A **TRANSACTION** may consist of many **RETURNS**. Each **RETURN** can only be part of a single **TRANSACTION**.

Each **RETURN** can consist of multiple **ITEMS** that are returned. An **ITEM** can be a part of many **RETURNS**. A bridge entity called **RETURN_ITEM** was created to address this relationship.

Finally, A **DELIVERY** must be shipped to a single **CUSTOMER**. A **CUSTOMER** may receive multiple **DELIVERIES**.

From this, we came up with the following entities:

CUSTOMER	**TRANSACTION**	**STORE**	**EMPLOYEE**
ADMINISTRATOR	**ITEM**	**CATEGORY**	**TRANSACTION_ITEM**
DELIVERY	**DELIVERY_ITEM**		

Entities with relation to Data Sources

Entities and how they relate the Data Sources	Data Sources…
Entity: Transaction For each item that is returned, the data source comes from the customer. The customer gives data about why they returned the item. Information about a specific customer is recorded each time a new customer wants to make a purchase from the store.	Customer
Entity: Store For the entity store, the data source would be the administrator. The main	Administrator

administrator owns the store and therefore chooses where he wants the store to be located and what phone number or StoreId number he wants.	
Entity: Transaction For the transaction entity, bulk purchase from the supplier will affect the price per unit. If more quantities are purchased for a specific item, the amount of discount will be proportional to how much quantity is purchased.	Supplier
Entity: Category For the category entity, it will affect the supplier list. The category entity contains the stock number, which allows us to determine if we need more stocks from our supplier.	Supplier
Entity: Administrator In the administrator entity, the data source would be customers. The customer gives information about themselves, email, phone number, name information, and they then make an online account.	Customers Suppliers
Entity: Delivery In the delivery entity, the data source would be the customer. The customer needs to give us their address information, name, so that the item can be delivered to the proper address with the correct name.	Customer
Entity: Item For the item entity, the data source would be the administrator. The administrator sets the price, the item number and the made/expiry date is set by the supplier.	Administrator Supplier
Entity: Delivery Item The data source for the delivery item would be the customer. The customer sets the quantity that they want and that amount is then delivered to the customer.	Customer
Entity: Transaction Item The data source for the transaction item is the customer. The customer himself sets the quantity that they want to purchase and the transaction is then completed with the amount requested by the customer. Bridge between Transaction and Item to create a M:N relationship.	Customer
Entity: Employee The data source for the employee is the employee themselves. The employee gives their information such as their name, birthdate, phone number, email so the system can store their information.	Employee

Sample data

```
SELECT * FROM Customer
```

CUSTOMERID	FIRSTNAME	LASTNAME	DOB	GENDER	PHONE	EMAIL	ADDRESS	POINTS	ISVIP	ISVALID
CU150001	Homer	Simpson	09/01/1990	1	6045561234	hsimpson@gmail.com	515 Ontario Street, Vancouver, BC V4X 1A7	-	0	1
CU150002	Tina	Smith	01/02/1	0	7785580	tsmith@gmail	21 Maroal	-	0	1

			980		230	.com	Dr., Richmond, BC V1N 3Y7			
CU150003	Alex	Spotter	02/02/1970	1	6041561234	aspotter@hotmail.com	4147 Chatham, Vancouver, BC V3E 5Y7	-	0	1
CU150004	Mara	Harper	09/10/1962	0	6047761234	mharper@netmail.com	777 Johns Cr., Surrey, BC V1B 2H4	-	0	1
CU150005	Ken	Dope	10/05/1959	1	6048880234	kdope@gmail.com	332 Chequeen Street, Coquitlam, BC V3E 0B9	-	0	1

SELECT * FROM **Employee**

EMPLOYEENO	FIRSTNAME	LASTNAME	DOB	GENDER	PHONE	EMAIL	ADDRESS	SIN	LV	POSITION	EDUCATION	STORENO	ISVALID
00001	Kim	Bell	09/10/1978	1	6047771234	kbell@gmail.com	110 Dupeen Dr. Coquitlam, BC V3E 0B2	777123456	3	Store Manager	MBA	SC-01	1
00002	Anna	Johns	01/08/1975	0	6047761238	ajohns@sports.com	3135 Silver St. Burnaby, BC V1X 7G1	701123455	2	Store Manager	MBA	SC-02	1
00003	Tony	Chueng	02/10/1971	1	7787751234	tchueng@gmail.com	666 Kimpon Cr. Landley, BC V9Y 3H1	713123450	2	Administrator	Electronic Engineer	SC-01	1
00004	Benedict	Singh	01/20/1989	1	6057799234	bsingh@netmail.com	445 Jump Road. Richmond, BC V7Y 3U3	775023456	2	Administrator	CST	SC-02	1
00005	Gordon	Broinne	03/04/1989	1	6047771200	gbroinne@gmail.com	900 Victory Road. Vancouver, BC V1E 2B2	773993456	1	Clerk	High School	SC-03	1

SELECT * FROM **Administrator**

ADMINID	EMPLOYEENO	USERNAME	PASSWD	ISVALID
1	00003	scadmin	202cb962ac59075b964b07152d234b70	1
2	00004	scadmin2	202cb962ac59075b964b07152d234b70	1

SELECT * FROM **Store**

STORENO	ADDRESS	PHONE
SC-01	2929 Barnet Highway #1400, Coquitlam, BC V3B 5R9	6044645110
SC-02	4700 Kingsway, Burnaby, BC V5H 4M1	6044645112
SC-03	18 West Broadway, Vancouver, BC V5Y 1P2	6044645113
SC-04	8125 Ontario Street, Vancouver, BC V5X 0A7	6044645114
SC-05	Unit 600 5771 Marine Way, Burnaby, BC V5J 0A6	6044645115

SELECT * FROM **Category**

CATNO	CATNAME	STOCKNO
CA150001	Apparel	N01
CA150002	Outwear	N01
CA150003	Equipment	N02
CA150004	Hockey	N02
CA150005	Footwear	N03

SELECT * FROM **Item**

ITEMNO	CATNO	ITEMNAME	PRICE	QUANTITY	PRODUCER	ENTRYDATE	MADEDATE	EXPIRYDATE
IT150001	CA150001	Columbia Hanath Mens Half-zip Top	23.99	50	Hanath	10/21/2015	01/18/2015	-
IT150002	CA150002	Giro Bevel White Helmet	53.99	50	Giro	10/21/2015	01/18/2015	-
IT150003	CA150003	Nordica Nadvo Binding	220.99	25	Nordica	10/21/2015	02/01/2015	-
IT150004	CA150004	Bauer Supreme Stick Flex	43.99	10	Bauer	10/21/2015	03/15/2015	-
IT150005	CA150005	Reebok Realflex Womens Boots	105.99	8	Reebok	10/21/2015	09/02/2015	-

SELECT * FROM **Transaction**

TRANSACTIONID	RECEIPTNO	PAYMENTNO	CUSTOMERID	TOTALCOST	STORENO	CASHIERNO	TRANSACTIONDATE	SUCCEED
TP00312345	SC11512345	VISA-7289011123	CU150001	299	SC-01	00005	10/21/2015	1
TP00312346	SC11512346	MAST-1179011124	-	199	SC-01	00001	10/21/2015	1
TP00312347	SC115123	MAST-	CU150003	399.5	SC-02	00002	10/21/2015	1

	47	3459011123						
TP00312348	SC11512348	VISA-7287011181	CU150004	1099	-	-	10/21/2015	1
TP00312349	SC11512349	VISA-3389011177	CU150005	700.1	-	-	10/21/2015	1

SELECT * FROM **Delivery**

DELIVERYID	TRANSACTIONID	SHIPPER	CONSIGNEE	DELIVERYDATE
DP01012345	TP00312348	00005	CU150004	10/21/2015
DP01012346	TP00312349	00005	CU150005	10/21/2015

SELECT * FROM **Transaction Item**

Output Grid 1 Environment

TRANSACTIONID	ITEMNO	QUANTITY
TP00312345	IT150001	2
TP00312346	IT150003	1
TP00312347	IT150005	2
TP00312348	IT150004	3
TP00312349	IT150002	4

SELECT * FROM **Delivery Item**

Output Grid 1 Environment

DELIVERYID	ITEMNO	QUANTITY
DP01012345	IT150004	3
DP01012346	IT150005	2

Relationships

Relationships and how they relate to the user stories	Data Sources…
For each customer, there can be many deliveries. But for each delivery made, it can only go to one customer. The relationship between the customer and transaction would be a 1:M relationship.	This relates to user story #4. Jermaine (Employee) wants to be able to check if customers have received their delivered products so the customer can be satisfied.
For each transaction, each one transaction can have more than one delivery. The relationship between the transaction and the delivery is a 1:M relationship.	This relates to user story #5. John (administrator) wants the item to be delivered right after the transaction is made so customers can receive their product as soon as

	possible.
For each category that we have in our store, the one category can have many items. Each item can only belong to one category. Therefore, this is a 1:M relationship.	This relates to user story #6. Bob (Employee) wants all the basketball and hockey items to go under their own specific shelf so customers can find the items more easily and not get confused when looking for specific equipment.
For each delivery made, there can be more than one item to be delivered. An item be a part of many deliveries. Therefore, we have a M:M relationship.	This relates to user story #7. When James(employee) makes a delivery to a household, he usually delivers more than one item at a time so no more than one trip would be have to made to the same location at a time.
For each customer, there can be many transactions made. This gives us a 1:M relationship.	This relates to user story #8. When Jamie (customer) wants to purchase items from the sports store for his work, he needs separate receipts to be able to claim back his reimbursement.
For each transaction, there can be more than one item, while the item can have many transactions as well. Therefore, the relationship is a M:N relationship.	This relates to user story #10. Harold (Employee) wants customers to purchase their items in a single transaction so no confusion or disorganization occurs.
An administrator is an employee, and an employee can be an administrator, but not all employees are administrators. Thus, we have a 1:1 relationship.	This relates to user story #11. Hassan (Employee) always reports back to his manager (Kobe) before he leaves work so he can form a better relationship with his boss.
Each store can have many employees, but those same employees can only be working at one store. Therefore, this is a 1:M relationship.	This relates to user story #12. Henry (Customer) wants to always have someone there to help him purchase footwear, so he always asks Jenny (Employee) who is the regular employee that knows his foot size and what he needs.

Data Dictionary

Table: Customer			
Column	**Type**	**Nullable**	**Description**
customerId	char(8)	not null	Primary key
firstName	varchar(15)	not null	First name
lastName	varchar(15)	not null	Last name
dob	date		Date of birth

Formatted Table

gender	char(1)		Gender: 0-female, 1-male
phone	varchar(10)	not null	Phone number
email	varchar(20)	not null	Email address
address	varchar(70)	not null	Home address
points	int		Bonus
isVip	char(1)		Is VIP? 0-No, 1-Yes, default-0
isValid	char(1)		Is Valid? 0-No, 1-Yes, default-1
Table: Employee			
Column	**Type**	**Nullable**	**Description**
employeeNo	char(5)	not null	Primary key
firstName	varchar(15)	not null	First name
lastName	varchar(15)	not null	Last name
dob	date		Date of birth
gender	char(1)		Gender: 0-female, 1-male
phone	varchar(10)	not null	Phone number
email	varchar(20)	not null	Email address
address	varchar(70)	not null	Home address
sin	char(9)	not null	Social insurance number
level	smallint	not null	Level. default – 1
position	varchar(15)	not null	Position.
education	varchar(30)		Education background
storeNo	char(5)		Foreign key, references to Store(storeNo). Null for High level managers
isValid	char(1)		Is Valid? 0-No, 1-Yes, default-1
Table: Administrator			
Column	**Type**	**Nullable**	**Description**
~~adminId~~	~~int~~	~~not null~~	~~Primary key.~~
employeeNo	char(5)	not null	Primary Key, Foreign key, references to Employee(employeeNo)
userName	varchar(10)	not null	Login name
passwd	varchar(32)	not null	Login password (MD5)
isValid	char(1)		Is Valid? 0-No, 1-Yes, default-1
Table: Item			
Column	**Type**	**Nullable**	**Description**
itemNo	char(8)	not null	Primary key
catNo	char(8)	not null	Foreign key, references to Category(catNo)
itemName	varchar(30)	not null	Item name
costPrice	Decimal(6,2)		Price. eg. 99.99. default -0.00
~~price~~salePrice	decimal(6,2)		Price. eg. 99.99. default -0.00
quantity	int		Quantity of items in stock. default-0
producer	varchar(30)		Producer
spec	varchar(30)		Item spec
entryDate	date		Entry date
madeDate	date		Made date

Formatted Table

Formatted Table

expiryDate	date		Expiry date
Table: Category			
Column	**Type**	**Nullable**	**Description**
catNo	char(8)	not null	Primary key
catName	varchar(20)	not null	Category name
stockNo	char(3)		Place in stock
Table: Transaction			
Column	**Type**	**Nullable**	**Description**
transactionId	char(10)	not null	Primary key
receiptNo	char(10)	not null	receipt number
paymentNo	varchar(15)	not null	Payment number. eg VISA-012334567
customerId	char(8)		Customer Id, foreign key. Non-registered customer's Id can be null
~~totalCost~~	~~decimal(6,2)~~		~~Total cost. Default 0.00~~
storeNo	char(5)		Foreign key, references to Store(storeNo). NULL for online purchase
cashierNo	char(5)		Cashier No, foreign key (employeeNo). Null for online purchase
transactionDate	date	not null	Transaction date
succeed	char(1)		Is transaction completed? 0-fail, 1-succeed
Table: Delivery			
Column	**Type**	**Nullable**	**Description**
deliveryId	char(10)	not null	Primary key
transactionId	char(10)	not null	Foreign key, references to Transaction(transactionId)
shipper	char(5)	not null	Shipper, foreign key Employee(employeeNo)
consignee	char(8)	not null	Consignee, foreign key Customer(customerId)
deliveryDate	date	not null	Delivery date
receiveDate	date		Received date
Table: Transaction_Item			
Column	**Type**	**Nullable**	**Description**
transactionId	char(10)	not null	Primary key, foreign key, references to Transaction(transactionId)
itemNo	char(8)	not null	Primary key, foreign key, references to Item(itemNo)
quantity	int		The quantity of items purchased
Table: Delivery_Item			
Column	**Type**	**Nullable**	**Description**
deliveryId	char(10)	not null	Primary key, foreign key, references to Delivery(deliverId)
itemNo	varchar(15)	not null	Primary key, foreign key, references to Item(itemNo)
quantity	int		The quantity of items shipped
Table: Store			

Formatted Table

Column	Type	Nullable	Description
storeNo	char(5)	not null	Primary key, eg. SC-01, SC-02
address	varchar(70)	not null	Store address
phone	varchar(10)	not null	Store phone number

Tables:

- Customer – Lists the registered customers
- Employee – Lists the employees who work for Sports Chek
- Administrator – Lists the administrators who manage the database system
- Item – Shows the items in stock
- Category – Shows the categories that the items belong to
- Transaction – Shows the transactions made online or in-store
- Return – Shows the returns that are made
- Delivery – Shows the deliveries shipping to the customers
- Transaction_Item – Bridge table that connects Transaction and Item
- Delivery_Item – Bridge table that connects Delivery and Item
- Return_Item – Bridge table that connects Return and Item
- Store – Lists the branch stores of Sports Chek

Track changes (For supplier list)

We added 3 tables in this milestone:
1).Store (storeId, address, phone).
Lists the branch stores of the Sports Shop. Employee table has a nullable column/attribute (storeNo) which references to Store(storeNo), Transaction also has a nullable column/attribute (storeNo) references to Store(storeNo).
2).Transaction_Item (transactionId, itemNo, quantity)
Bridge entity that connects Transaction and Item to allow M:N relationship.
3).Delivery_Item (deliveryId, itemNo, quantity)
Bridge table that connects Delivery and Item to allow a M:N relationship.

As per further requirement analysis, most sport stores use fixed id length for some columns, such as customerId, employeeNo, transactionId, deliveryId, etc. Therefore, we made minor changes to the data types of the following columns:

Column	Old Type	New Type
customerId	int	char(8), *eg.11512345*
employeeNo	int	char(5), *eg. SC1234*
transactionId	int	char(10), *eg. TP00312345*
deliveryId	int	char(10), *eg. DP01012345*

Milestone 2

Part: A - Detailed Descriptions of Stakeholders

Stakeholder #1: Store Owner / Store Manager

~~**What kind of information do they need?**~~
~~Remaining inventory~~
~~Rate of sales (So they know that they should bring more stock next time)~~
~~List of ordered items with arrival times~~

~~**How often do they need the information?**~~
~~At most once or twice a day, ensure healthy stock left in store.~~

~~**What do they want to do with the data?**~~
~~Ensure that an item is hopefully never depleted (sold out)~~
~~Identify sales trends immediately, such as a **HOT** item that has been selling well.~~
~~Ensure that ordered items arrive on time, and to make space in-store for them as well.~~

~~**Summary descriptions of information required from the database (entity names?)**~~
~~*Remaining inventory:*~~
~~item_id, stock~~
~~*Rate of sales:*~~
~~Sold_Today (using SQL sum() AS Sold_Today GROUP BY itemName)~~

The first stakeholder we have would be the Store Owner / Manager. Due to the fact that both of them are technically in charge of overseeing the store, they therefore have the same amount of power in the Database. Some information they would need to know the overall remaining inventory of the store, rate of sales (To determine approximately how much stock to order) and a list of currently ordered items with respective arrival times.

They will most likely only require the information once or twice a day, to ensure that everything is well stocked, and the currently sold-out items have stock on its way to the store. By using the database effectively and efficiently, they can keep track of an items status, so that an item is never depleted or sold out. They can also identify sale trends immediately, so that a **HOT** item that has been selling well is always in stock for customers to purchase. Lastly, they can also utilize the arrival times to ensure that an item is never sold out, and to make sure that there is plenty of room to store and display new arrivals.

Stakeholder #2: Employees (Working in-store)

~~**What kind of information do they need?**~~
~~Remaining inventory~~
~~Customer information to process a sale~~
~~Customer information to process a return~~
~~Maintenance services~~

~~**How often do they need the information?**~~

~~Quite often, depending on amount of customers they have to deal with in a single day. Almost any interaction with a customer will require them to do access the DB.~~

~~**What do they want to do with the data?**~~
~~Check if there is remaining stock in the back to be placed on the sales floor~~
~~Process a sale / return~~
~~Take in an item for maintenance~~

~~**Summary descriptions of information required from the database**~~
~~*Remaining inventory:*~~
~~item_id, stock~~
~~*Customer information to process a sale:*~~
~~fName, lName, email, address, phoneNum, customer_ID~~
~~*Customer information to process a return:*~~
~~receipt_ID, item_ID, itemName, dateReturn, price, customer_ID, returnReason~~
~~*Maintenance services:*~~
~~maint_Type, maint_ID, maint_Time, dateRecieved, dateTime(How long it takes), datePickup, customer_ID~~

Formatted: Space After: 0 pt

The second stakeholder are the employees. The information that the employees require would be knowing the remaining inventory, Information of an item to process a sale/return, and the procedure of maintenance services on items which are brought in for tune-ups.

For an average employee, they will most likely require constant access to the database depending on the amount of customers they have to deal with during their work shift. Almost any interaction with a customer will likely require them to pull up information on a certain item, or to process a sale. Some things that they will use the database for would be to check if there is any remaining stock left in storage that can be put on the sales floor, processing a sale/return, and recording down customer information for an item taken in for tune-ups.

Formatted: Space After: 0 pt

Stakeholder #3: Customers (Online orders)

~~**What kind of information do they need?**~~
~~customer_ID~~
~~ID of the item they wish to purchase.~~
~~stock~~

~~**How often do they need the information?**~~
~~Not very often, only when they feel like making an online purchase.~~

~~**What do they want to do with the data?**~~
~~Make an online purchase~~

~~**Summary descriptions of information required from the database**~~
~~*Customer_ID: (Note: If they have purchased something from the store before, an ID will be issued for them already. All they need to do is associate their remaining information to create an online account which has a customer_ID that goes with it.)*~~
~~fName, lName, email, address, phoneNum, birthDate, gender~~

~~*ID of the item they wish to purchase:*~~
~~Item_ID~~

~~*Remaining inventory:*~~
~~item_id, stock~~

The last stakeholder would be the Customers themselves, to process an online order. The information a customer needs would be their customer ID which is tied with all of their credentials, the item number they wish to purchase, and the item's stock.

A customer will most likely not require information very often, only requiring it when making an online purchase. By checking if an item is available online, they can directly purchase their item online or by reserving an item in store to pick it up at a convenient time.

User Stories (Only 1 Key Point or Purpose / User Story)

~~1) Kenny (Store Owner) finds that with the data of returnReason, it helps him know what was wrong with the product and whether they need change or not.~~
~~2) Ben (Store Manager) uses the price key which helps him determine the profits for any particular item in store.~~
~~3) Barry (Store Manager) finds that with the use of phoneNum, it allows him to make any calls to customers regarding follow-up on any complaints~~
~~4) Jermaine (Employee) finds that with the use of dateReceived, it allows him to see whether the customer has received their online purchase yet.~~
~~5) Jerry (Employee) finds that with the use of customerID, it allows him to see how many online purchases were made for the current month.~~
~~6) Jaimie (Employee) finds that by using the email key, it allows her to email the customer if they have not picked up their item in-store that was purchased online.~~
~~7) Jenny (Employee) finds that by using the stock key, it really helps with knowing when to purchase more inventory.~~
~~8) John (Employee) finds that by using the itemName really helps speed up the process of a sale when there are long lineups.~~
~~9) James (Employee) finds that by having a receiptID, it really helps speed up the process of a return.~~
~~10) Henry finds that by using the datePickup, it helps with knowing when to pick up her online purchase in-store.~~
~~11) Harold (Customer) uses the stock key to let himself know whether there are any more items in stock left for him to purchase~~
~~12) Harry (Customer) finds that by using the itemID it helps speed up the process to find a specific item.~~
~~13) Johnny (Customer) finds that the by using the itemID, the webpage may notify him of similar items at a better price.~~

1) Kenny (Store Owner) wants to know why people return the product so he knows which products to stop selling.
2) Ben (Store Manager) wants to find out the profits for each item so he knows what products to advertise more.
3) Jaimie (Employee) wants customer's emails to alert them that their item has arrived in store.
4) Jermaine (Employee) wants to be able to check if customers have received their delivered products so the customer can be satisfied.
5) John (administrator) wants the item to be delivered right after the transaction is made so customers can receive their product as soon as possible.
6) Bob (Employee) wants all the basketball and hockey items to go under their own specific shelf so customers can find the items more easily and not get confused when looking for specific equipment

7) James (employee) wants to track the batch deliveries (at least 3 items ordered at a time).
8) After purchases, Jamie (customer) wants to check points in his account.
9) Jerry (customer) wants to make many purchases at the sports store, so for convenience, he always uses the store located closest to his house.
10) Harold (Employee) wants customers to purchase their items in a single transaction so no confusion or disorganization occurs.
11) Hassan (Employee) always reports back to his manager (Kobe) before he leaves work so he can form a better relationship with his boss.
12) Henry (Customer) wants to always have someone there to help him purchase footwear, so he always asks Jenny (Employee) who is the regular employee that knows his foot size and what he needs.

Formatted: Font: (Asian) Times New Roman, 10 pt, Font color: Text 2

Formatted: Normal, Indent: Left: 0.25", No bullets or numbering

Part: B – Data Requirements

Data sources

Customers – Information about the specific customer is recorded each time a new customer wishes to make a purchase from the store.
Supplier List – List of all the retailers which we purchase inventory from
Supplier Price – For determining prices of goods, which can be used to determine where to get the best deal.
Sport Teams – List of all players from a specific team with their respective sizes, colors, and designs of their team.

Data format

Sports shop database contains numerous amounts of data regarding in-store/online purchases. The corresponding data format (draft) can be depicted in the following tables:

Customer

Column	Type	Is Null	Description
customerId	int	not null	Primary key. Auto increment.
firstName	varchar(15)	not null	First name
lastName	varchar(15)	not null	Last name
dob	date		Date of birth
gender	char(1)		Gender: 0-female, 1-male
phone	varchar(10)	not null	Phone number
email	varchar(20)	not null	Email address
address	varchar(70)	not null	Home address
points	int		Bonus
isVip	char(1)		Is VIP? 0-No, 1-Yes, default-0
isValid	char(1)		Is Valid? 0-No, 1-Yes, default-1

Employee

Column	Type	Is Null	Description
employeeNo	int	not null	Primary key
firstName	varchar(15)	not null	First name
lastName	varchar(15)	not null	Last name

dob	date		Date of birth
gender	char(1)		Gender: 0-female, 1-male
phone	char(10)	not null	Phone number
email	varchar(20)	not null	Email address
address	varchar(70)	not null	Home address
sin	char(9)	not null	Social insurance number
level	tinyint	not null	Level. default – 1
position	varchar(15)	not null	Position.
education	varchar(30)		Education background
isValid	char(1)		Is Valid? 0-No, 1-Yes, default-1

Administrator

Column	Type	Is Null	Description
adminId	int	not null	Primary key.
employeeNo	int	not null	Foreign key
userName	varchar(10)	not null	Login name
passwd	varchar(10)	not null	Login password
isValid	char(1)		Is Valid? 0-No, 1-Yes, default-1

Item

Column	Type	Is Null	Description
itemNo	varchar(15)	not null	Primary key
catNo	varchar(15)	not null	Foreign key
itemName	varchar(30)	not null	Item name
price	decimal(6,2)		Price. eg. 99.99. default -0.00
amount	int		Amount. default-0
producer	varchar(30)		Producer
entryDate	date		Entry date
madeDate	date		Made date
expiryDate	date		Expiry date

Category

Column	Type	Is Null	Description
catNo	varchar(15)	not null	Primary key
catName	varchar(20)	not null	Category name
stockNo	varchar(3)		Place in stock

Transaction

Column	Type	Is Null	Description
transactionId	int	not null	Primary key, auto increment, start from a certain number. eg 115000000
receiptNo	varchar(15)	not null	receipt number
paymentNo	varchar(15)	not null	Payment number. eg VISA-012334567
customerId	int		Customer Id, foreign key. Non-registered customer's

			Id can be null
totalCost	decimal(6,2)		Total cost. Default 0.00
storeId	tinyint	not null	Store ID. 0-online, 1, 2, 3…
cashierNo	int		Cashier No, foreign key (employeeNo)
items	varchar(100)		Items purchased, consists of itemNo*amount, different items separated by comma. eg. NG-002*5, TD-101*3
transactionDate	date		Transaction date
succeed	char(1)		Is transaction completed? 0-fail, 1-succeed

Delivery

Column	Type	Is Null	Description
deliveryId	int	not null	Primary key, auto increment, start from a certain number. eg 115000000
transactionId	int	not null	Transaction ID, foreign key
receiptNo	varchar(15)	not null	Receipt No, foreign key
itemNo	varchar(15)	not null	Item No, foreign key
itemAmount	int		Item amount
shipper	int	not null	Shipper, foreign key (employeeNo)
consignee	int	not null	Consignee, foreign key (CustomerId)
deliveryDate	date		Delivery date

How the data is updated

Open an account: employees enter customer's information such as customer's first name, last name, phone number, email, and home address into the system (insert).

In-store purchase: customer Id, items purchased, prices, date of purchase, total cost, payment No, cashier Id, cashier name, store Id, and receipt No are recorded to the database (insert), meanwhile items purchased are deducted from the inventory (update the amount of the items) and the customer's points are updated as well.

Online purchases, order Id, customer Id, items purchased, prices, date of purchase, total cost, payment No, receipt No, shipping address, and delivery date are recorded to the database (insert), meanwhile items purchased are deducted from the inventory (update the amount of the items) and the customer's points are updated as well.

Stock in: entry No, entry items, date, delivery company, contact person, phone number, and address are recorded to the database (insert), meanwhile the amount of entry items in stock are updated.

Stock out: shipping No, shipping items, shipping address, customer name, and date are recorded to the database (insert), meanwhile the amount of entry items in stock are updated.

Return/refund: receipt No, returning items, date, prices, customer Id, and reason are recorded to the database (insert), meanwhile the amount of returning items in stock are updated.

Computations / calculations performed on the data

Some basic computations / calculations such as SUM(price), COUNT(*), price * discount percentage, adding redeemed points, etc. will be performed on the database frequently.

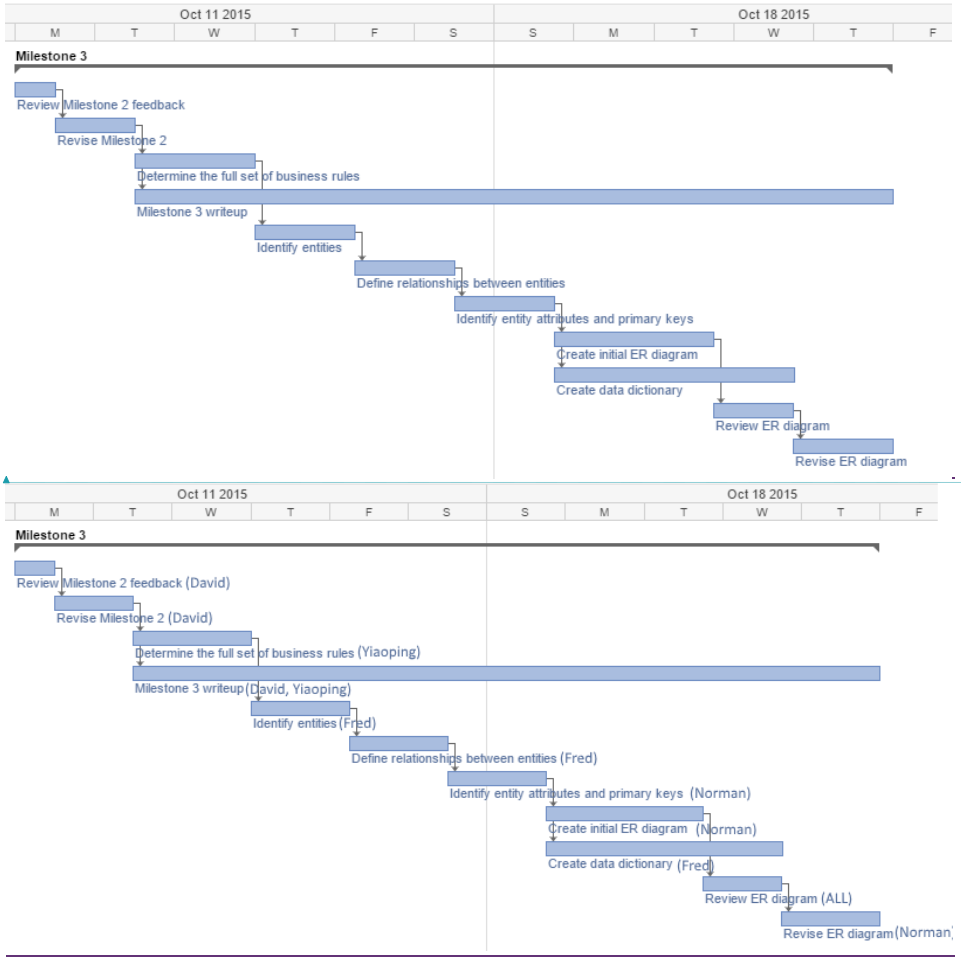
Part: C – Project Planning

Since most of us are now learning the relational database systems from the beginning and hardly have experience, basically we will do all together at first by helping and teaching each other.

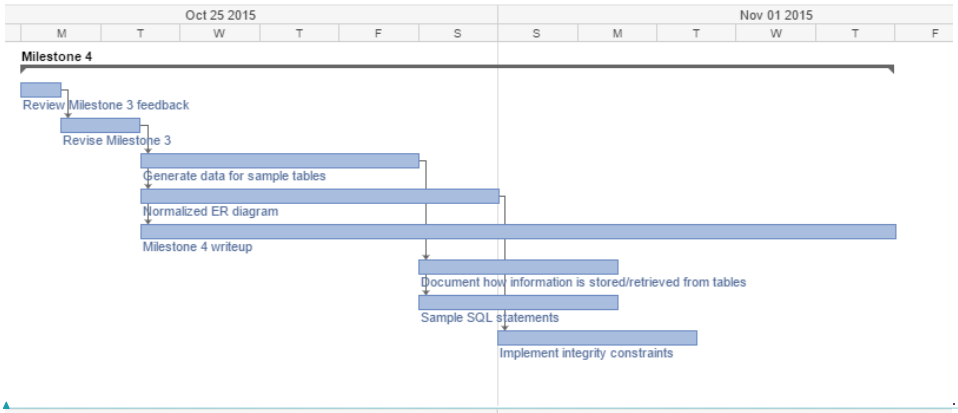
Here are the ways we're planning to work on this project.

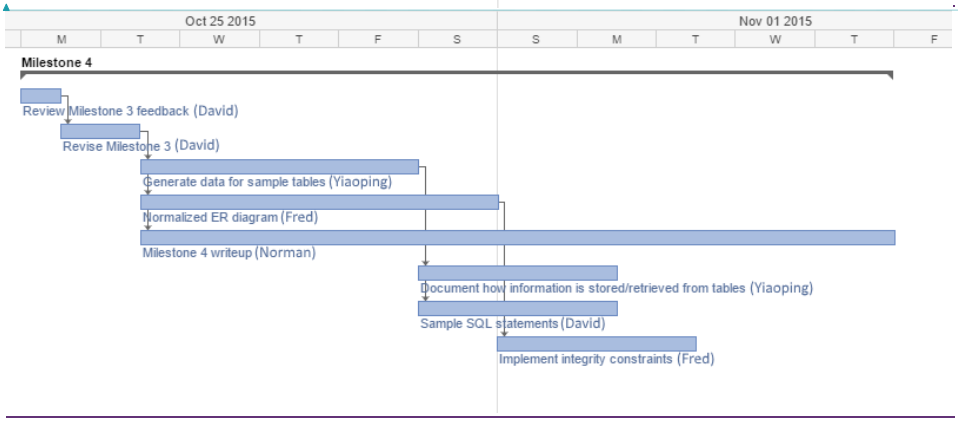
- Use Gantt Chart to plan and track the project.
- Schedule the meeting at least once every week.
- Communicate in real time by group-chatting on Facebook.
- Bring at least one idea individually to every meeting and discuss together.
- Divide the work into small pieces for each step at every meeting.
- Assign the work equally as per each members' strengths in this project.
- Put priority on this team project over other individual's assignments or work before the deadline is due.
- Help each other to manage to meet deadlines.

Gantt Chart:

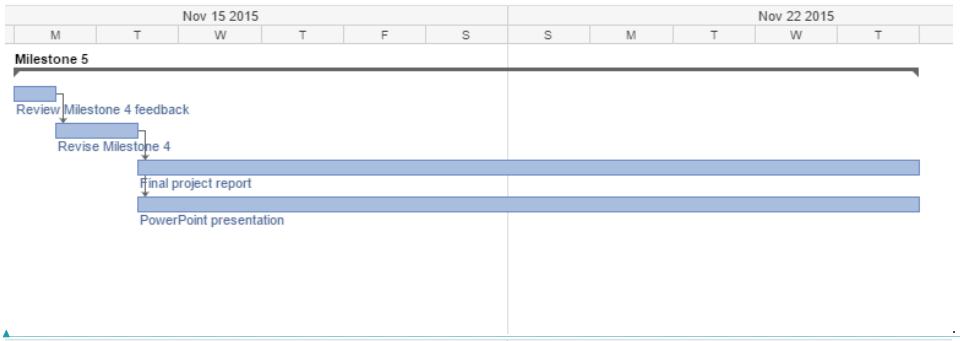


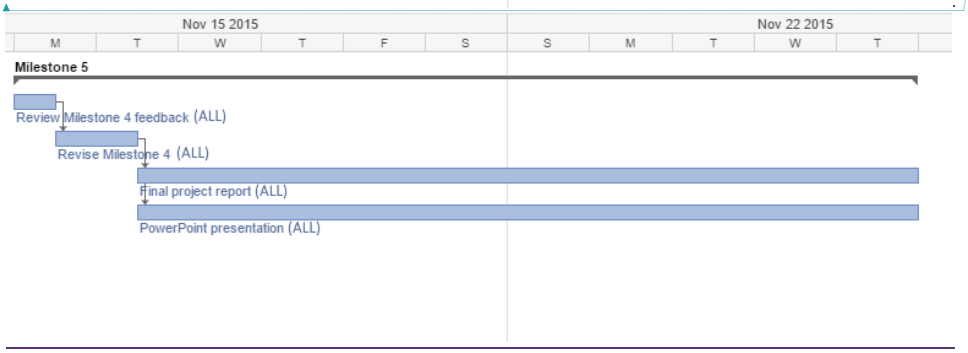
Formatted: Font: (Asian) Times New Roman, 10 pt, Font color: Text 2





Formatted: Font: (Asian) Times New Roman, 10 pt, Font color: Text 2





Formatted: Font: 10 pt, Font color: Text 2

Milestone 1

Database Project Proposal

Sports Shop Database (Option 1)

1. Brief background information

a. Context

Sports Shop Inventory Database

Designed for processing daily sales, list of shipments, and inventory management.

b. Stakeholders

Store owner, employees, and customers

c. Benefits of such a system (e.g. allows tracking, helps salesperson to answer queries)

- Quick inventory check
- Easy to keep track if something is low on stock
- Purchases that are scanned out are immediately deducted from the database.
- Keep track of customer records by having them create accounts.
- Online purchasing of goods
- Better estimates of completion / arrival times on orders or maintenance services.

2. Requirement description. A sample requirement description for an Order Entry and Shipment System could read:

Anyone who wishes to purchase from the store will require an account. Employees can easily assign a default account instantly for a new customer which only requires their first + last name, and their phone number. When a customer purchases items from the sports store, the database updates the inventory count, deducting it from the database. The purchase is logged, and the items purchased, date of purchase, total cost, cashier name, and receipt number are recorded to the database. If the customer was to return the item, an employee would be able to confirm that it is a legitimate return by looking up the information in the database using the receipt number, and the employee would be able to process the return, which would add the item back onto the database. The database also acts as a way of checking remaining inventory in the store. Customers can link their default accounts at the stores website, which then allows them to make any online purchases. Any purchases made both online and in-store will have e-receipts which are then saved in our database through their online accounts. Our database will also keep track of how many online purchases and in-store purchases are made. Online purchases will also record the shipping address, as well as the shipping date.

Norman Lim, Fred Yang, David Yu, Yiao Shu COMP 2714

Database Project Proposal

Library Database (Option 2)

1. Brief background information

a. Context

School Library Information Management System Database

Designed for processing the daily borrowing, returning, and various query options of library information.

b. Stakeholders

Students, Librarians

c. Benefits of such a system (e.g. allows tracking, helps salesperson to answer queries)

- Accurate tracking of entire library catalog
- Book information management (ISBN, title, category, author, editions, press, year, price, format, stock, assigned branch, modifications, deletion, etc.)
- General information query: Books information, Status of book (borrowed or on shelf)
- User management: Readers library card id, name, address, birthdate, phone, email, currently borrowed, date returned, outstanding books and fees, etc.
- Staff management: almost everything above, with inclusion of additional powers to directly alter database records.
- System settings: user login and pw

2. Requirement description. A sample requirement description for an Order Entry and Shipment System could read:

A student can borrow up to 10 books, from any of the institution's libraries. All regular books can be borrowed for up to 14 days. However, all course related textbooks can be only borrowed for up to 24 hours. Students can firstly sign-up on the school libraries website to create an account by using their school student id. Students can then log in to the libraries website with their id to check information of their books such as: currently borrowed, due dates, overdue books, book reservations and outstanding fees. Outstanding fees will restrict the respective account to borrow any more books until the fees are paid off. Overdue fees can be paid off by cash only, at the help desk of any institutions. Librarians are responsible for checking in books in the returned books drop box, and aiding in records management. Records management include updating records in the database such as adding or removing books, and aiding students in settling any conflicts with their respective accounts.