# A Database Management System

# for

# Books-A-Thousand Bookstores

CSC 540 Database Management concepts and Systems

Project Report #2

Team Rdm
Abhay Gupta(agupta15), Rohit Kandhal(rkandha), Sagar Jauhari(sjauhar), Raghav Tripathi(rtripat)

## List of assumptions taken for Books-A-Thousand store (new)

1. We don't have multiple editions of a book/merchandise. Each name and author uniquely determines a merchandise
2. If an employee wants to make a purchase from the store then he should be added as new customer.
3. An employee can be employed at only one store at a time.
4. There can be no anonymous authors
5. Status of an order can be CANCELLED, DELIVERED, SHIPPED and NOT SHIPPED
6. Selling price of an item can be zero when it is offered as a free item
7. Every staff person has exactly one job title/role
8. Name and phone number of the people record uniquely identifies other attributes of the people entity.
9. Each vendor has a unique name
10. Payment for vendor is done on the 1st month which is the billing cycle.

## List of assumptions taken for Books-A-Thousand store (previous report)

1. Warehouse of the bookstore has merchandise purchased only from the vendors.
2. Each merchandise is supplied to the store by only one vendor.
3. Books-A-Thousand buys merchandise (i.e. warehouse stock) only from vendors who are under contract.
4. Store has salesperson who helps customers with merchandise selection and once customer selects, salesperson can swipe that merchandise for further order processing.
5. Billing cycle for each vendor and each customer is first day of month.
6. There is only one warehouse for the complete chain of Books-A-Thousand stores.
7. Age of a person (staff, customer) is not kept as a separate attribute but calculated from date of birth.
8. Isolation level for the database system is set to serializable.
9. When a particular order is shipped by the vendor then we assume that order will be received safely by the store and warehouse database will be updated with the merchandises received.
10. Merchandise purchased from a vendor is sold at some profit by Books-A-Thousand thus prices of a merchandise sold by vendor and sold by Books-A-Thousand store may vary.
11. Purchased made by a customer can be assisted by atmost one salesperson.
12. Usage record for vendor or customer is not specifically maintained as a separate entity in database. All orders placed by customer or vendor will be in customer order or vendor orders, and after successful completion of an order these would be used to retrieve usage required.
13. Merchandise ID in the warehouse will remain same as that shipped by the vendor i.e. No two vendors can use the same merchandise ID for their books.
14. A store doesn't maintain any inventory of its own instead all merchandise are fetched from the central warehouse.

## 1) Database Schema

a. **vendors(<u>id</u>, name, address, phone_number)**
vendor_id->name , address,phone_number
name->vendor_id,address,phone_number
Both vendor_id and name can be primary key, hence this table is in BCNF so in 3NF.

b. **Merchandise(<u>mer_id</u>, c_price,name,author,book_info,vendor_id)**
mer_id->c_price,name,author,book_info,vendor_id
name,author->book_info,price,mer_id,vendor_id

Name of the merchandize and author together taken are unique.As mer_id and (name,author) both are primary key hence table in 3NF.

c. **vendor_payments(<u>id</u>,payment_date,amount,vendor_id)**
id->vendor_id,payment_date,amount

Table already in 3NF because only one FD having RHS as a primary key

d. **vendor_order(<u>id</u>,status,order_date,cost,vendor_id)**
id->status,order_date,cost (Table in BCNF,hence in 3NF)

e. **vendor_order_contents(<u>order_id</u>, mer_id,quantity)**
order_id -> vendor_id,quantity (Table in BCNF,hence in 3NF)

f. **warehouse_contents(<u>mer_id</u>,discount, selling_price, qty_in_stock)**
mer_id->discount,selling_price,qty_in_stock (Table in BCNF,hence in 3NF)

g. **customer_order_contents(<u>mer_id</u>,<u>order_id</u>,quantity)**
order_id ,mer_id-> mer_id,order_id,quantity (Table in BCNF,hence in 3NF)

h. **customer_order(<u>id</u>,order_date,status,sales_person_pid,customer_id)**
id->order_date,status,sales_person_pid,customer_id (Table in BCNF,hence in 3NF)

i. **ordered_by(<u>order_id</u>,sales_person_pid,pid_cust)**
order_id -> pid_staff, pid_cust (Table in BCNF,hence in 3NF)

j. **people(<u>pid</u>, name, ssn, gender, address, phone, date_of_birth)**
pid->name, ssn, gender, address, phone, date_of_birth
name,phone->gender,date_of_birth,address,ssn

No FD for ssn because it can be null for one or more people hence does not uniquely determines anything.Name and phone number together is a primary key hence the relationship does not violate primary key.

k. **store (<u>id</u>, short_name, address)**
id -> address
short_name -> address
short_name is another primary key of this relationship hence the schema is in 3NF

l. **staff(<u>pid</u>, job_title, dept_name, store_id)**
pid->job_title, dept_name, store_id (Table in BCNF,hence in 3NF)

m. **customer(<u>pid</u>, balance, is_preferred_cust)**
pid->balance, is_preferred_cust  (Table in BCNF,hence in 3NF)

n. **cust_payments(<u>id</u>, cust_id, payment_amount, payment_date)**
id-> cust_id, payment_amount, payment_date   (Table in BCNF,hence in 3NF)

## 2) Design Decisions

A mechanical approach was used to create the global schema.

a. Each entity set was made into a relation with the same set of attributes
b. Relationships were replaced by a relation whose attributes are the keys for the connected entity sets

E-R approach was used to convert subclass of into relations. The relationship having Many-to-one association were combined with other relations so as to give more efficient answer to queries.

**vendors(<u>id</u>, name, address, phone_number)**
id (primary key): unique identifier
name (not null, unique) : every vendor has got to have some name

**merchandise(<u>id</u>, name, author, price, book_info, vendor_id)**
id (primary key) : unique identifier of the merchandise
name (not null, unique) : a particular book has to be written by one and only one author
author (not null): every book as an author
price (not null): every merchandize has some price
book_info : information maintained about every book like Fiction, Non Fiction, Fantasy etc
vendor_id:(referential integrity) : refers to the vendors table vendor_id

**vendor_payments(<u>id</u>, payment_date, amount, vendor_id)**
id (primary key): uniquely identifies a payment made to the vendor
payement_date (not null ) : kept as not null to as to maintain a log and generate reports
amount (not null): keep track of the payment made
vendor_id (referential integrity) : refers to the vendor table (vendor_id) to whom payment is made

**vendor_order(<u>id</u>, status, order_date, cost, vendor_id)**
id (primary key): unique identifier of the order placed
status (not null): Possible values are CANCELLED, DELIVERED, SHIPPED, NOT SHIPPED status of the order is always well defined at any instance hence not null
order_date (not null) : keeps track of the date when the order was placed
cost (not null): keeps track of the cost for which the payment was made
vendor_id (referential integrity ): refers to the vendor table

**vendor_order_contents(<u>order_id</u>,<u>mer_id</u>,quantity)**
order_id(primary key, referential integrity): refers to the vendor_id of the vendor_order table
mer_id(primary key,referential integrity) : refers to the mer_id of the merchandize table
quantity(not null,check constraint >0): quantity of the order

**warehouse_contents(<u>mer_id</u>,discount, selling_price, qty_in_stock)**
mer_id (primary key, referential integrity): refers to the mer_id of the merchandize table
selling_price(not null,check constraint >=0): selling price of the merchandize which cannot be null
qty_in_stock(not null,check constraint >=0): identifies the stock of an item,zero if none present

**customer_order_contents(<u>mer_id</u>,<u>order_id,</u>quantity)**
mer_id(primary key,referential integrity): refers to the mer_id of the warehouse_contents table as the order can be placed only of the available items in the bookstore
order_id (primary key,referential integrity): refers to the order_id of the cust_order table(orders made by the customer)
quantity(not null,check constraint >0): identifies the stock of an item,zero if none present

**customer_order(<u>id</u>,order_date,status,sales_person_pid,customer_id)**
id(primary key): uniquely identifies order placed by the customer
order_date(not null): date at which customer placed an order
status(not null) : Possible values are CANCELLED, DELIVERED, SHIPPED, NOT SHIPPED status of an order at a particular instant
sales_person_pid(referential integrity) : refers to the pid of the staff table
customer_id(referential integrity): refers to the pid of the customer table

**ordered_by(<u>order_id</u>,sales_person_pid,pid_cust)**
order_id (primary key,referential integrity): refers to the order_id of the cust_order table(orders made by the customer)

**people(<u>pid</u>, date_of_birth, name, gender, address, phone, ssn)**
pid (primary key): unique identifier of a person
name(not null): name of the person
date_of_birth(not null): date of birth of a person
gender(not null): gender of a person
address(not null): address information of a person
phone:can be null for a person
ssn (unique): can be null for a person
(name,phone): unique together

**staff(<u>pid</u>, job_title, dept_name,store_id)**
pid(primary key, referential integrity):refers to the people table
job_title (not null): designation of a staff
dept_name(not null):should not be null
store_id(referential integrity): refers to the store_id of the store table

**customer(<u>pid</u>, is_preferred_cust, balance)**
pid(primary key, referential integrity):refers to the people table
is_prefered_cust(check constraint either 0 or 1) : check for preferred customer
balance(not null): outstanding balance of the customer; zero if no balance

**cust_payments(i<u>d</u>,cust_id,payment_amount,payment_date)**
id (primary key): uniquely identifies a payment made by the customer
cust_id(primary key,referential integrity):refers to the pid of the customer
payment_amount(not null): keep track of the payment made
payment_date(not null ) : kept as not null to as to maintain a log and generate reports

**store (<u>id</u>,short_name,address)**
id(primary key):uniquely identifies a store
short_name(not null,unique):has a unique name for the store
address(not null): identifies the address of the store

## 3) SQL Statements

### Drop Tables

```
DROP TABLE vendors CASCADE CONSTRAINTS;
DROP TABLE merchandise CASCADE CONSTRAINTS;
DROP TABLE vendor_payments CASCADE CONSTRAINTS;
DROP TABLE vendor_order CASCADE CONSTRAINTS;
DROP TABLE vendor_order_contents CASCADE CONSTRAINTS;
DROP TABLE warehouse_contents CASCADE CONSTRAINTS;
DROP TABLE customer_order_contents CASCADE CONSTRAINTS;
DROP TABLE customer_order CASCADE CONSTRAINTS;
DROP TABLE people cascade constraints;
DROP TABLE store cascade constraints;
DROP TABLE staff cascade constraints;
DROP TABLE customer cascade constraints;
DROP TABLE cust_payments cascade constraints;
```

## Create Tables

```
CREATE TABLE vendors (
    id RAW(16) PRIMARY KEY,
    name VARCHAR2(64) NOT NULL UNIQUE,
    address VARCHAR2(128),
    phone_number VARCHAR2(20)
);

CREATE TABLE merchandise (
    id RAW(16) PRIMARY KEY,
    name VARCHAR2(64) NOT NULL UNIQUE,
    author VARCHAR2(64) NOT NULL,
    price NUMBER NOT NULL,
    book_info VARCHAR2(128),
    vendor_id RAW(16),
    CONSTRAINT mer_vendor_id_fk
            FOREIGN KEY (vendor_id)
            REFERENCES vendors (id)
            ON DELETE CASCADE,
        CONSTRAINT mer_pri_con
            CHECK (price > 0)
);

CREATE TABLE vendor_payments (
    id RAW(16) PRIMARY KEY,
    payment_date DATE NOT NULL,
    amount NUMBER NOT NULL,
        vendor_id RAW(16),
    CONSTRAINT vendor_payments_vendor_id_fk
        FOREIGN KEY (vendor_id)
        REFERENCES vendors (id)
            ON DELETE CASCADE,
        CONSTRAINT ven_pay_amt_cons
            CHECK (amount > 0)
);
```

```
CREATE TABLE vendor_order (
        id RAW(16) PRIMARY KEY,
        status VARCHAR2(20) NOT NULL,
        order_date DATE NOT NULL,
        cost NUMBER NOT NULL,
        vendor_id RAW(16),
        CONSTRAINT vendor_order_vendor_id_fk
                        FOREIGN KEY (vendor_id)
                        REFERENCES vendors (id)
                        ON DELETE CASCADE,
                CONSTRAINT ven_cost_cons
                        CHECK (amount > 0),
                CONSTRAINT customer_order_status
                        CHECK (status='DELIVERED' OR status='CANCELLED' OR status='SHIPPED' OR status='NOT
                SHIPPED')
);

CREATE TABLE vendor_order_contents (
        order_id RAW(16),
        mer_id RAW(16),
        quantity NUMBER NOT NULL,
        CONSTRAINT order_id_fk
                FOREIGN KEY (order_id)
                REFERENCES vendor_order (id)
                ON DELETE CASCADE,
        CONSTRAINT mer_id_fk
                FOREIGN KEY (mer_id)
                REFERENCES merchandise (id)
                ON DELETE CASCADE,
        CONSTRAINT vendor_order_contents_qty
                CHECK (quantity>0),
        PRIMARY KEY(order_id, mer_id)
);
```

```
CREATE TABLE warehouse_contents (
        mer_id RAW(16),
        selling_price VARCHAR2(64) NOT NULL,
        qty_in_stock VARCHAR2(64) NOT NULL,
        discount NUMBER,
        CONSTRAINT mer_id_warehouse_contents_fk
                FOREIGN KEY (mer_id)
                REFERENCES merchandise (id)
                ON DELETE CASCADE,
        CONSTRAINT warehouse_order_contents_qty
                CHECK (qty_in_stock >=0),
        CONSTRAINT warehouse_order_contents_price
                CHECK (selling_price >=0),
        PRIMARY KEY(mer_id)
);

CREATE TABLE customer_order_contents (
        order_id RAW(16),
        mer_id RAW(16),
        quantity NUMBER NOT NULL,
        CONSTRAINT coc_order_id_fk
                FOREIGN KEY (order_id)
                REFERENCES customer_order (id)
                ON DELETE CASCADE,
        CONSTRAINT coc_mer_id_fk
                FOREIGN KEY (mer_id)
                REFERENCES merchandise (id)
                ON DELETE CASCADE,
        CONSTRAINT coc_qty
                CHECK (quantity>0),
        PRIMARY KEY(order_id, mer_id)
);
```

```
CREATE TABLE customer_order (
	id RAW(16) PRIMARY KEY,
	status VARCHAR2(20) NOT NULL,
	order_date DATE  NOT NULL,
	sales_person_pid RAW(16) NOT NULL,
	customer_pid RAW(16) NOT NULL,
	CONSTRAINT co_person_pid_fk
		FOREIGN KEY (sales_person_pid )
		REFERENCES people(pid)
		ON DELETE CASCADE,
	CONSTRAINT co_customer_pid_fk
		FOREIGN KEY (customer_pid )
		REFERENCES people(pid)
		ON DELETE CASCADE,
	CONSTRAINT customer_order_status
	CHECK (status='DELIVERED' OR status='CANCELLED' OR status='SHIPPED' OR status='NOT SHIPPED')
);

CREATE TABLE people (
	pid RAW(16) PRIMARY KEY,
	name VARCHAR(64) NOT NULL,
	ssn VARCHAR(20) UNIQUE,
	gender VARCHAR2(64) NOT NULL,
	address VARCHAR2(128) NOT NULL,
	phone VARCHAR2(20),
	date_of_birth DATE NOT NULL,
	UNIQUE (name, phone)
);

CREATE TABLE store(
	id RAW (16) PRIMARY KEY,
	short_name VARCHAR2(64) NOT NULL UNIQUE,
	address VARCHAR2(128) NOT NULL
);
```

```
CREATE TABLE staff (
        pid RAW(16) PRIMARY KEY,
        job_title VARCHAR2(64) NOT NULL,
        dept_name VARCHAR2(64) NOT NULL,
        store_id RAW(16) NOT NULL,
        CONSTRAINT staff_pid_fk
                FOREIGN KEY (pid)
                REFERENCES people(pid)
                ON DELETE CASCADE,
        CONSTRAINT staff_store_id_fk
                FOREIGN KEY (store_id)
                REFERENCES store(id)
                ON DELETE CASCADE
);

CREATE TABLE customer (
        pid RAW(16) PRIMARY KEY,
        balance NUMBER DEFAULT 0 NOT NULL,
        is_preferred_cust NUMBER(1,0) DEFAULT 0 CHECK (is_preferred_cust IN (1,0)),
        CONSTRAINT customer_pid_fk
                FOREIGN KEY (pid)
                REFERENCES people(pid)
                ON DELETE CASCADE
);

CREATE TABLE cust_payments(
        id RAW(16) PRIMARY KEY,
        cust_id RAW(16) NOT NULL,
        payment_amount NUMBER DEFAULT 0,
        payment_date DATE DEFAULT sysdate ,
        CONSTRAINT payments_pid_fk
                FOREIGN KEY (cust_id)
                REFERENCES customer(pid)
                ON DELETE CASCADE
);
```

## Inserts

```
INSERT INTO vendors VALUES (SYS_GUID(), 'Harper Collins', '5126 Bur Oak Cir, Raleigh, NC', '919-666-7777');
INSERT INTO vendors VALUES (SYS_GUID(), 'Schiel and Denver', '5122 Bur Oak Cir, Raleigh, NC', '919-555-6666');
INSERT INTO vendors VALUES (SYS_GUID(), 'Random House', '3101 Hillsborough St, Raleigh, NC', '919-444-5555');
INSERT INTO vendors VALUES (SYS_GUID(), 'Ivy House', '3618 Cannold Ct, Raleigh, NC', '919-111-2222');

INSERT INTO merchandise VALUES (SYS_GUID(), 'Sweet Tooth', 'Ian McEwan', '16.17', 'Fiction', (SELECT id FROM vendors where name='Harper Collins'));
INSERT INTO merchandise VALUES (SYS_GUID(), 'Proof of Heaven', 'Eben Alexander', '9.35', 'Non Fiction', (SELECT id FROM vendors where name='Harper Collins'));
INSERT INTO merchandise VALUES (SYS_GUID(), 'Dear Life: Stories', 'Alice Munro', '16.17', 'Non Fiction', (SELECT id FROM vendors where name='Schiel and Denver'));
INSERT INTO merchandise VALUES (SYS_GUID(), 'The Racketeer', 'John Grisham', '16.53', 'Fiction', (SELECT id FROM vendors where name='Schiel and Denver'));
INSERT INTO merchandise VALUES (SYS_GUID(), 'Days of Blood and Starlight', 'Laini Taylor', '11.39', 'Fantasy', (SELECT id FROM vendors where name='Random House'));
INSERT INTO merchandise VALUES (SYS_GUID(), 'Cloud Atlas', 'David Mitchell', '8.78', 'Fantasy', (SELECT id FROM vendors where name='Random House'));
INSERT INTO merchandise VALUES (SYS_GUID(), 'Hallucinations', 'Oliver Sacks', '15.85', 'Fiction', (SELECT id FROM vendors where name='Ivy House'));
INSERT INTO merchandise VALUES (SYS_GUID(), 'Gone Girl', 'Gillian Flynn', '13.94', 'Fiction', (SELECT id FROM vendors where name='Ivy House'));
INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2011/10/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '805', (SELECT id FROM vendors where name='Harper Collins'));
INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2011/11/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '1600', (SELECT id FROM vendors where name='Harper Collins'));
INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2011/12/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '2560', (SELECT id FROM vendors where name='Harper Collins'));
INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2012/01/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '1750', (SELECT id FROM vendors where name='Harper Collins'));
INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2012/02/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '350', (SELECT id FROM vendors where name='Harper Collins'));
INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2012/03/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '455', (SELECT id FROM vendors where name='Harper Collins'));
INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2012/04/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '1030', (SELECT id FROM vendors where name='Harper Collins'));
INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2012/05/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '1210', (SELECT id FROM vendors where name='Harper Collins'));
INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2012/06/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '1600', (SELECT id FROM vendors where name='Harper Collins'));
INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2012/07/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '1200', (SELECT id FROM vendors where name='Harper Collins'));
INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2012/08/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '1705', (SELECT id FROM vendors where name='Harper Collins'));
INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2012/09/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '525', (SELECT id FROM vendors where name='Harper Collins'));
INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2012/10/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '715', (SELECT id FROM vendors where name='Harper Collins'));
INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2012/10/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '850', (SELECT id FROM vendors where name='Schiel and Denver'));
INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2012/09/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '340', (SELECT id from vendors where name='Schiel and Denver'));
INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2012/08/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '450', (SELECT id from vendors where name='Schiel and Denver'));
INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2012/10/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '950', (SELECT id from vendors where name='Random House'));
INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2012/09/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '420', (SELECT id from vendors where name='Random House'));
INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2012/08/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '1350', (SELECT id from vendors where name='Random House'));
INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2012/10/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '1050', (SELECT id from vendors where name='Ivy House'));
```

```
INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2012/09/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '250', (SELECT id from vendors where name='Ivy House'));
INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2012/08/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '1400', (SELECT id from vendors where name='Ivy House'));

INSERT INTO vendor_order VALUES (SYS_GUID(), 'NOT SHIPPED', TO_DATE('2012/10/01 08:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '715', (SELECT id from vendors where name='Harper Collins'));
INSERT INTO vendor_order VALUES (SYS_GUID(), 'SHIPPED', TO_DATE('2012/10/01 08:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '850', (SELECT id from vendors where name='Schiel and Denver'));
INSERT INTO vendor_order VALUES (SYS_GUID(), 'SHIPPED', TO_DATE('2012/10/01 08:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '950', (SELECT id from vendors where name='Random House'));
INSERT INTO vendor_order VALUES (SYS_GUID(), 'DELIVERED', TO_DATE('2012/10/01 08:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '1050', (SELECT id from vendors where name='Ivy House'));

INSERT INTO vendor_order_contents VALUES(
        (SELECT id from vendor_order where vendor_id =(SELECT vendor_id from merchandise WHERE
        name='Sweet Tooth')),
        (SELECT id from merchandise WHERE name='Sweet Tooth'), 1);

INSERT INTO vendor_order_contents VALUES(
        (SELECT id from vendor_order where vendor_id =(SELECT vendor_id from merchandise WHERE
        name='Dear Life: Stories')),
        (SELECT id from merchandise WHERE name='Dear Life: Stories'), 1);

INSERT INTO vendor_order_contents VALUES(
        (SELECT id from vendor_order where vendor_id =(SELECT vendor_id from merchandise WHERE
        name='Days of Blood and Starlight')),
        (SELECT id from merchandise WHERE name='Days of Blood and Starlight'), 1);

INSERT INTO vendor_order_contents VALUES(
        (SELECT id from vendor_order where vendor_id =(SELECT vendor_id from merchandise WHERE
        name='Hallucinations')),
        (SELECT id from merchandise WHERE name='Hallucinations'), 1);
```

```
INSERT INTO warehouse_contents VALUES ((SELECT id from merchandise WHERE name='Sweet Tooth'), 1000, 2,
3);

INSERT INTO warehouse_contents VALUES ((SELECT id from merchandise WHERE name='Dear Life: Stories'), 2000,
2, 3);

INSERT INTO warehouse_contents VALUES ((SELECT id from merchandise WHERE name='Days of Blood and
Starlight'), 3000, 2, 3);

INSERT INTO warehouse_contents VALUES ((SELECT id from merchandise WHERE name='Hallucinations'), 4000, 2,
3);

INSERT INTO customer_order_contents
VALUES((SELECT id from customer_order where sales_person_pid=(SELECT pid FROM people WHERE name='John'
AND phone='919-111-1111')), (SELECT id from merchandise WHERE name='Sweet Tooth'), 1);

INSERT INTO customer_order_contents VALUES(
        (SELECT id from customer_order where sales_person_pid=(SELECT pid FROM people WHERE
        name='Joseph' AND phone='919-111-2222')),
        (SELECT id from merchandise WHERE name='Dear Life: Stories'), 2);

INSERT INTO customer_order_contents VALUES(
        (SELECT id from customer_order where sales_person_pid=(SELECT pid FROM people WHERE
        name='Michael' AND phone='222-333-1111')),
        (SELECT id from merchandise WHERE name='Days of Blood and Starlight'), 4);

INSERT INTO customer_order_contents VALUES(
        (SELECT id from customer_order where sales_person_pid=(SELECT pid FROM people WHERE name='Andy'
        AND phone='555-111-5555')),
        (SELECT id from merchandise WHERE name='Hallucinations'), 1);
```

```
INSERT INTO customer_order VALUES (
        SYS_GUID(),
        'NOT SHIPPED', (SELECT to_date(to_char(sysdate,'yyyy/mm/dd hh24:mi:ss'),'yyyy/mm/dd hh24:mi:ss')
        FROM dual),
        (SELECT pid FROM people WHERE name='John' AND phone='919-111-1111'),
        (SELECT pid FROM people WHERE name='Jack' AND phone='445-243-9876')
);

INSERT INTO customer_order VALUES (
        SYS_GUID(), 'NOT SHIPPED',  (SELECT to_date(to_char(sysdate,'yyyy/mm/dd hh24:mi:ss'),'yyyy/mm/dd
        hh24:mi:ss') FROM dual),
        (SELECT pid FROM people WHERE name='Joseph' AND phone='919-111-2222'),
        (SELECT pid FROM people WHERE name='Lucy' AND phone='445-243-9876')
);

INSERT INTO customer_order VALUES (SYS_GUID(), 'NOT SHIPPED',  (SELECT to_date(to_char(sysdate,'yyyy/mm/
dd hh24:mi:ss'),'yyyy/mm/dd hh24:mi:ss') FROM dual),
        (SELECT pid FROM people WHERE name='Michael' AND phone='222-333-1111'),
        (SELECT pid FROM people WHERE name='Abraham' AND phone='445-243-9876')
);

INSERT INTO customer_order VALUES (SYS_GUID(), 'NOT SHIPPED',
        (SELECT to_date(to_char(sysdate,'yyyy/mm/dd hh24:mi:ss'),'yyyy/mm/dd hh24:mi:ss') FROM dual),
        (SELECT pid FROM people WHERE name='Andy' AND phone='555-111-5555'),
        (SELECT pid FROM people WHERE name='Narla' AND phone='445-243-9876')
);

INSERT INTO customer_order VALUES (SYS_GUID(), 'NOT SHIPPED',
        (SELECT to_date(to_char(sysdate,'yyyy/mm/dd hh24:mi:ss'),'yyyy/mm/dd hh24:mi:ss') FROM dual),
        (SELECT pid FROM people WHERE name='Jim' AND phone='444-222-3333'),
        (SELECT pid FROM people WHERE name='Narla' AND phone='445-243-9876')
);

INSERT INTO store values (SYS_GUID(), 'Avent Ferry', '1 Avent Ferry Shopping Complex, Raleigh, NC');
INSERT INTO store values (SYS_GUID(), 'Hillsborough', '989 Hillsborough Street, Raleigh, NC');
INSERT INTO store values (SYS_GUID(), 'NewYork', 'Tower 1, Times Square,New York');
INSERT INTO store values (SYS_GUID(), 'California', '1007, Caltech shopping complex, California');
INSERT INTO store values (SYS_GUID(), 'Delhi', '219 Cannought Place, New Delhi, India');
INSERT INTO people VALUES ( SYS_GUID(), 'John', '1234567', 'Male', '123 CREST RD, RALEIGH, NC', '919-111-1111',
TO_DATE('1985/01/01 16:01:00', 'yyyy/mm/dd hh24:mi:ss'));
INSERT INTO people VALUES ( SYS_GUID(), 'Joseph', '1234589', 'Male', '123 HILLS, RALEIGH, NC', '919-111-
2222',TO_DATE('1981/01/01 16:01:00', 'yyyy/mm/dd hh24:mi:ss'));
INSERT INTO people VALUES ( SYS_GUID(), 'Michael', '1231234', 'Male', '123 AVENT FERRY RD, RALEIGH, NC', '222-
333-1111',TO_DATE('1987/01/01 16:01:00', 'yyyy/mm/dd hh24:mi:ss'));
INSERT INTO people VALUES ( SYS_GUID(), 'Andy', '1253432', 'Male', 'MOUNT ABU, Rajesthan, India', '555-111-
5555',TO_DATE('1984/01/01 16:01:00', 'yyyy/mm/dd hh24:mi:ss'));
INSERT INTO people VALUES ( SYS_GUID(), 'Jim', '1234789', 'Male', '1621 Times Square, Newyork', '444-222-
3333',TO_DATE('1983/01/01 16:01:00', 'yyyy/mm/dd hh24:mi:ss'));
INSERT INTO people VALUES ( SYS_GUID(), 'Fodu', '1432123', 'Male', '23, Nai Sarak, Old Delhi, India', '555-666-
7777',TO_DATE('1980/01/01 16:01:00', 'yyyy/mm/dd hh24:mi:ss'));
INSERT INTO people VALUES ( SYS_GUID(), 'Sam', '1238783', 'Male', '2838, Fernor Street, Allentown, PA', '666-444-
2222',TO_DATE('1984/01/01 01:01:00', 'yyyy/mm/dd hh24:mi:ss'));
INSERT INTO people VALUES ( SYS_GUID(), 'Nick', '1432323', 'Male', '2312, Champion Court, Raleigh, NC', '764-234-
1234',TO_DATE('1984/01/01 01:01:00', 'yyyy/mm/dd hh24:mi:ss'));
INSERT INTO people VALUES ( SYS_GUID(), 'Chotu', '1487623', 'Male', '1233, Champion Court, Raleigh, NC', '445-
243-2112',TO_DATE('1984/01/01 01:01:00', 'yyyy/mm/dd hh24:mi:ss'));
INSERT INTO people VALUES ( SYS_GUID(), 'Christie' , '1568923', 'Female', '1233, Champion Court, Raleigh,
NC', '445-243-2111',TO_DATE('1987/01/01 01:01:00', 'yyyy/mm/dd hh24:mi:ss'));
INSERT INTO people VALUES ( SYS_GUID(), 'Lipu' , '1433453', 'Female', '2512, Avery Close, Raleigh, NC', '445-243-
9876',TO_DATE('1987/01/01 01:01:00', 'yyyy/mm/dd hh24:mi:ss'));
INSERT INTO people VALUES ( SYS_GUID(), 'Jack', '14212453', 'Male', '2512, Avery Close, Raleigh, NC', '445-243-
9877',TO_DATE('1987/01/01 01:01:00', 'yyyy/mm/dd hh24:mi:ss'));
INSERT INTO people VALUES ( SYS_GUID(), 'Lucy' , '14323435', 'Female', '2512, Avery Close, Raleigh, NC', '445-243-
9878',TO_DATE('1987/01/01 01:01:00', 'yyyy/mm/dd hh24:mi:ss'));
INSERT INTO people VALUES ( SYS_GUID(), 'Abraham' , '6433453', 'Female', '2512, Avery Close, Raleigh, NC', '445-
243-9879',TO_DATE('1987/01/01 01:01:00', 'yyyy/mm/dd hh24:mi:ss'));
INSERT INTO people VALUES ( SYS_GUID(), 'Narla' , '9333453', 'Female', '2512, Avery Close, Raleigh, NC', '445-243-
9880',TO_DATE('1987/01/01 01:01:00', 'yyyy/mm/dd hh24:mi:ss'));
INSERT INTO people VALUES ( SYS_GUID(), 'Smith' , '8333453', 'Female', '2512, Avery Close, Raleigh, NC', '445-243-
9881',TO_DATE('1987/01/01 01:01:00', 'yyyy/mm/dd hh24:mi:ss'));
```

```
INSERT INTO staff VALUES ((SELECT pid FROM people p WHERE p.Name= 'John' AND p.phone = '919-111-
1111'), 'Sales Assistant', 'Sales', (SELECT id FROM store WHERE short_name = 'NewYork'));
INSERT INTO staff VALUES ((SELECT pid FROM people p WHERE p.Name= 'Joseph' AND p.phone = '919-111-
2222'), 'Cashier', 'Sales', (SELECT id FROM store WHERE short_name = 'NewYork'));
INSERT INTO staff VALUES ((SELECT pid FROM people p WHERE p.Name= 'Michael' AND p.phone = '222-333-
1111'), 'Sales Assistant', 'Sales', (SELECT id FROM store WHERE short_name = 'NewYork'));
INSERT INTO staff VALUES ((SELECT pid FROM people p WHERE p.Name= 'Andy' AND p.phone = '555-111-
5555'), 'Manager', 'Management', (SELECT id FROM store WHERE short_name = 'Delhi'));
INSERT INTO staff VALUES ((SELECT pid FROM people p WHERE p.Name= 'Jim' AND p.phone = '444-222-
3333'), 'Assistant', 'Management', (SELECT id FROM store WHERE short_name = 'Delhi'));

INSERT INTO customer VALUES ((SELECT pid FROM people p WHERE p.name = 'Sam' AND p.phone = '666-444-
2222'), 0, 1);
INSERT INTO customer VALUES ((SELECT pid FROM people p WHERE p.name = 'Nick' AND p.phone = '764-234-
1234'), 0, 1);
INSERT INTO customer VALUES ((SELECT pid FROM people p WHERE p.name = 'Chotu' AND p.phone = '445-243-
2112'), 0, 0);
INSERT INTO customer VALUES ((SELECT pid FROM people p WHERE p.name = 'Christie' AND p.phone = '445-243-
2111'), 0, 0);
INSERT INTO customer VALUES ((SELECT pid FROM people p WHERE p.name = 'Lipu' AND p.phone = '445-243-
9876'), 0, 0);
INSERT INTO customer VALUES ((SELECT pid FROM people p WHERE p.name = 'Jack' AND p.phone = '445-243-
9877'), 0, 1);
INSERT INTO customer VALUES ((SELECT pid FROM people p WHERE p.name = 'Lucy' AND p.phone = '445-243-
9878'), 0, 0);
INSERT INTO customer VALUES ((SELECT pid FROM people p WHERE p.name = 'Abraham' AND p.phone = '445-243-
9879'), 0, 1);
INSERT INTO customer VALUES ((SELECT pid FROM people p WHERE p.name = 'Narla' AND p.phone = '445-243-
9880'), 0, 1);

INSERT INTO cust_payments VALUES (SYS_GUID(), (SELECT pid FROM people p WHERE p.name = 'Chotu' AND
p.phone = '445-243-2112'), 200, TO_DATE('2012/03/04 01:01:00', 'yyyy/mm/dd hh24:mi:ss'));
INSERT INTO cust_payments VALUES (SYS_GUID(), (SELECT pid FROM people p WHERE p.name = 'Chotu' AND
p.phone = '445-243-2112'), 200, TO_DATE('2012/04/05 01:01:00', 'yyyy/mm/dd hh24:mi:ss'));
INSERT INTO cust_payments VALUES (SYS_GUID(), (SELECT pid FROM people p WHERE p.name = 'Lucy' AND
p.phone = '445-243-9878'), 100, TO_DATE('2012/05/06 01:01:00', 'yyyy/mm/dd hh24:mi:ss'));
INSERT INTO cust_payments VALUES (SYS_GUID(), (SELECT pid FROM people p WHERE p.name = 'Lucy' AND
p.phone = '445-243-9878'), 20, TO_DATE('2012/02/07 01:01:00', 'yyyy/mm/dd hh24:mi:ss'));
INSERT INTO cust_payments VALUES (SYS_GUID(), (SELECT pid FROM people p WHERE p.name = 'Lucy' AND
p.phone = '445-243-9878'), 30, TO_DATE('2012/03/08 01:01:00', 'yyyy/mm/dd hh24:mi:ss'));
```

```
INSERT INTO cust_payments VALUES (SYS_GUID(), (SELECT pid FROM people p WHERE p.name = 'Sam' AND
p.phone = '666-444-2222'), 40, TO_DATE('2012/02/09 01:01:00', 'yyyy/mm/dd hh24:mi:ss'));
INSERT INTO cust_payments VALUES (SYS_GUID(), (SELECT pid FROM people p WHERE p.name = 'Sam' AND
p.phone = '666-444-2222'), 50, TO_DATE('2012/02/10 01:01:00', 'yyyy/mm/dd hh24:mi:ss'));
```

## Selects

Note: Font size of query results has been decreased for better formatting and improved readability

SQL> SELECT * FROM vendors;

```
ID                                                           NAME                    ADDRESS
PHONE_NUMBER
----------------------------------------------   ------------------------   ------------------------------------------   ---------------------
CDA04D6ABAFB1990E0400E98195649FB    Harper Collins           5126 Bur Oak Cir, Raleigh, NC              919-666-
7777
CDA04D6ABAFC1990E0400E98195649FB    Schiel and Denver     5122 Bur Oak Cir, Raleigh, NC              919-555-
6666
CDA04D6ABAFD1990E0400E98195649FB    Random House         3101 Hillsborough St, Raleigh, NC       919-444-
5555
CDA04D6ABAFE1990E0400E98195649FB     Ivy House                3618 Cannold Ct, Raleigh, NC              919-111-
2222
```

SQL> SELECT * FROM vendor_order;

```
ID                                                    STATUS           ORDER_DATE  COST      VENDOR_ID
-----------------------------------------------   -----------------  ----------------     ----     ----------------------------------------------
CDA089BB3C7DFA08E0400E9819564ACD   NOT SHIPPED   01-10-12            715
   CDA04D6ABAFB1990E0400E98195649FB
CDA089BB3C7EFA08E0400E9819564ACD   SHIPPED           01-10-12            850
   CDA04D6ABAFC1990E0400E98195649FB
CDA089BB3C7FFA08E0400E9819564ACD   SHIPPED           01-10-12            950
   CDA04D6ABAFD1990E0400E98195649FB
CDA089BB3C80FA08E0400E9819564ACD   DELIVERED       01-10-12          1050
   CDA04D6ABAFE1990E0400E98195649FB
```

SQL> SELECT * FROM vendor_payments;

```
ID                                   PAYMENT_DATE   AMOUNT    VENDOR_ID
------------------------------------ -------------- --------- ------------------------------------
CDA07C78655A840DE0400E9819564AA0  01-10-11      805     CDA04D6ABAFB1990E0400E98195649FB
CDA07C78655B840DE0400E9819564AA0  01-11-11     1600     CDA04D6ABAFB1990E0400E98195649FB
CDA07C78655C840DE0400E9819564AA0  01-12-11      2560    CDA04D6ABAFB1990E0400E98195649FB
CDA07C78655D840DE0400E9819564AA0  01-01-12      1750    CDA04D6ABAFB1990E0400E98195649FB
CDA07C78655E840DE0400E9819564AA0  01-02-12      350
       CDA04D6ABAFB1990E0400E98195649FB
CDA07C78655F840DE0400E9819564AA0  01-03-12      455
       CDA04D6ABAFB1990E0400E98195649FB
CDA07C786560840DE0400E9819564AA0  01-04-12      1030    CDA04D6ABAFB1990E0400E98195649FB
CDA07C786561840DE0400E9819564AA0  01-05-12      1210    CDA04D6ABAFB1990E0400E98195649FB
CDA07C786562840DE0400E9819564AA0  01-06-12      1600    CDA04D6ABAFB1990E0400E98195649FB
CDA07C786563840DE0400E9819564AA0  01-07-12      1200    CDA04D6ABAFB1990E0400E98195649FB
CDA07C786564840DE0400E9819564AA0  01-08-12      1705    CDA04D6ABAFB1990E0400E98195649FB
CDA07C786565840DE0400E9819564AA0  01-09-12     525
       CDA04D6ABAFB1990E0400E98195649FB
CDA07C786566840DE0400E9819564AA0  01-10-12     715
       CDA04D6ABAFB1990E0400E98195649FB
CDA07C786567840DE0400E9819564AA0  01-10-12      850
       CDA04D6ABAFC1990E0400E98195649FB
CDA07C786568840DE0400E9819564AA0  01-09-12      340
       CDA04D6ABAFC1990E0400E98195649FB
CDA07C786569840DE0400E9819564AA0  01-08-12     450
       CDA04D6ABAFC1990E0400E98195649FB
CDA07C78656A840DE0400E9819564AA0  01-10-12     950
       CDA04D6ABAFD1990E0400E98195649FB
CDA07C78656B840DE0400E9819564AA0  01-09-12     420
       CDA04D6ABAFD1990E0400E98195649FB
CDA07C78656C840DE0400E9819564AA0  01-08-12      1350    CDA04D6ABAFD1990E0400E98195649FB
CDA07C78656D840DE0400E9819564AA0  01-10-12     1050     CDA04D6ABAFE1990E0400E98195649FB
CDA07C78656E840DE0400E9819564AA0  01-09-12      250
       CDA04D6ABAFE1990E0400E98195649FB
CDA07C78656F840DE0400E9819564AA0  01-08-12      1400    CDA04D6ABAFE1990E0400E98195649FB
```

```
SQL> SELECT * FROM merchandise;

ID                                               NAME                 AUTHOR           PRICE  BOOK_INFO      VENDOR_ID
------------------------------------------------ -------------------- ---------------- -------- ----------- -----------------------------------------
-----------
CDA068FC496F1B53E0400E9819564A71 Sweet Tooth          Ian McEwan      16.17 Fiction
   CDA04D6ABAFB1990E0400E98195649FB
CDA068FC49701B53E0400E9819564A71 Proof of Heaven   Eben Alexander 9.35   Fiction
   CDA04D6ABAFB1990E0400E98195649FB
CDA068FC49711B53E0400E9819564A71 Dear Life: Stories  Alice Munro      16.17 Fiction
   CDA04D6ABAFC1990E0400E98195649FB
CDA068FC49721B53E0400E9819564A71 The Racketeer      John Grisham    16.53 Fiction
   CDA04D6ABAFC1990E0400E98195649FB
CDA068FC49731B53E0400E9819564A71 Days of Blood      Laini Taylor      11.39 Fantasy
 CDA04D6ABAFD1990E0400E98195649FB
CDA068FC49741B53E0400E9819564A71 Cloud Atlas         David Mitchell    8.78 Fantasy
 CDA04D6ABAFD1990E0400E98195649FB
CDA068FC49751B53E0400E9819564A71 Hallucinations     Oliver Sacks      15.85 Fiction
   CDA04D6ABAFE1990E0400E98195649FB
CDA068FC49761B53E0400E9819564A71 Gone Girl            Gillian Flynn     13.94 Fiction
   CDA04D6ABAFE1990E0400E98195649FB
```

```
SELECT * FROM vendor_order_contents;
CDB2ECBE4EB1493EE0400E9819560546 CDA1C4B082CE97ABE0400E9819564BCB         1
CDB2ECBE4EB2493EE0400E9819560546 CDA1C4B082D097ABE0400E9819564BCB         1
CDB2ECBE4EB3493EE0400E9819560546 CDA1C4B082D297ABE0400E9819564BCB         1
CDB2ECBE4EB4493EE0400E9819560546 CDA1C4B082D497ABE0400E9819564BCB         1
```

```
SELECT * FROM warehouse_contents;
CDA1C4B082CE97ABE0400E9819564BCB 1000   2       3
CDA1C4B082D097ABE0400E9819564BCB 2000   2       3
CDA1D4B082D097ABE0400E9819564BCB 3000   2       3
CDA1C4B082D097ABE0400E9819564BCB 2000   2       3
CDA1C4B082D297ABE0400E9819564BCB 3000   2       3
CDA1C4B082D497ABE0400E9819564BCB 4000   2       3
```

```
SELECT * FROM customer_order_contents;
CDB2ECBE4EA5493EE0400E9819560546  CDA1C4B082CE97ABE0400E9819564BCB              1
CDB2ECBE4EA6493EE0400E9819560546  CDA1C4B082D097ABE0400E9819564BCB              2
CDB2ECBE4EA7493EE0400E9819560546  CDA1C4B082D297ABE0400E9819564BCB              4
CDB2ECBE4EA8493EE0400E9819560546  CDA1C4B082D497ABE0400E9819564BCB              1
```

SELECT * FROM customer_order;

```
CDB2ECBE4EA5493EE0400E9819560546 NOT SHIPPED          04-NOV-12
CDB2ECBE4E95493EE0400E9819560546 CDB2ECBE4EA0493EE0400E9819560546

CDB2ECBE4EA6493EE0400E9819560546 NOT SHIPPED          04-NOV-12
CDB2ECBE4E96493EE0400E9819560546 CDB2ECBE4EA1493EE0400E9819560546

CDB2ECBE4EA7493EE0400E9819560546 NOT SHIPPED          04-NOV-12
CDB2ECBE4E97493EE0400E9819560546 CDB2ECBE4EA2493EE0400E9819560546

CDB2ECBE4EA8493EE0400E9819560546 NOT SHIPPED          04-NOV-12
CDB2ECBE4E98493EE0400E9819560546 CDB2ECBE4EA3493EE0400E9819560546

CDB2ECBE4EAB493EE0400E9819560546 NOT SHIPPED          04-NOV-12
CDB2ECBE4E99493EE0400E9819560546 CDB2ECBE4EA3493EE0400E9819560546
```

SELECT * FROM store;

```
ID                                 SHORT_NAME    ADDRESS
------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------
CDB285DD0DF5AC4DE0400E98195604F3   Avent Ferry   1 Avent Ferry Shopping Complex, Raleigh, NC
CDB285DD0DF6AC4DE0400E98195604F3   Hillsborough  989 Hillsborough Street, Raleigh, NC
CDB285DD0DF7AC4DE0400E98195604F3   NewYork       Tower 1, Times Square,New York
CDB285DD0DF8AC4DE0400E98195604F3   California          1007, Caltech shopping complex, California
CDB285DD0DF9AC4DE0400E98195604F3   Delhi         219 Cannought Place, New Delhi, India
```

SELECT * FROM people;

```
PID                                   Name     ssn       gender   Address                              phone
date_of_birth
--------------------------------------------------------------------------------------------------------------------------------------------
---------------------
CDB285DD0E0AAC4DE0400E98195604F3  John    1234567  Male       123 CREST RD, RALEIGH, NC           919-111-1111         01-JAN-85
CDB285DD0E0BAC4DE0400E98195604F3  Joseph  1234589  Male       123 HILLS, RALEIGH, NC              919-111-2222         01-JAN-81
CDB285DD0E0CAC4DE0400E98195604F3  Michael 1231234  Male       123 AVENT FERRY RD, RALEIGH, NC            222-333-1111         01-
JAN-87
CDB285DD0E0DAC4DE0400E98195604F3  Andy    1253432  Male       MOUNT ABU, Rajesthan, India         555-111-5555         01-JAN-84
CDB285DD0E0EAC4DE0400E98195604F3  Jim     1234789  Male       1621 Times Square, Newyork          444-222-3333         01-JAN-83
CDB285DD0E0FAC4DE0400E98195604F3  Fodu    1432123  Male       23, Nai Sarak, Old Delhi, India     555-666-7777         01-JAN-80
CDB285DD0E10AC4DE0400E98195604F3  Sam     1238783  Male       2838, Fernor Street, Allentown, PA  666-444-2222         01-JAN-84
CDB285DD0E11AC4DE0400E98195604F3  Nick    1432323  Male       2312, Champion Court, Raleigh, NC   764-234-1234         01-JAN-84
CDB285DD0E12AC4DE0400E98195604F3  Chotu   1487623  Male       1233, Champion Court, Raleigh, NC   445-243-2112         01-JAN-84
CDB285DD0E13AC4DE0400E98195604F3  Christie 1568923 Female     1233, Champion Court, Raleigh, NC   445-243-2111         01-JAN-87
CDB285DD0E14AC4DE0400E98195604F3  Lipu    1433453  Female     2512, Avery Close, Raleigh, NC      445-243-9876         01-JAN-87
CDB285DD0E15AC4DE0400E98195604F3  Jack    14212453 Male       2512, Avery Close, Raleigh, NC      445-243-9877         01-JAN-87
CDB285DD0E16AC4DE0400E98195604F3  Lucy    14323435 Female     2512, Avery Close, Raleigh, NC      445-243-9878         01-JAN-87
CDB285DD0E17AC4DE0400E98195604F3  Abraham 6433453 Female      2512, Avery Close, Raleigh, NC      445-243-9879         01-JAN-87
CDB285DD0E18AC4DE0400E98195604F3  Narla   9333453 Female      2512, Avery Close, Raleigh, NC      445-243-9880         01-JAN-87
CDB285DD0E19AC4DE0400E98195604F3  Smith   8333453 Female      2512, Avery Close, Raleigh, NC      445-243-9881         01-JAN-87
```

SELECT * FROM staff;

```
PID                                   Job_title          Dept_name      Store_id
--------------------------------------------------------------------------------------------------------------------------------------------
---------------------
CDB285DD0E0AAC4DE0400E98195604F3      Sales Assistant    Sales          CDB285DD0DF7AC4DE0400E98195604F3
CDB285DD0E0BAC4DE0400E98195604F3      Cashier            Sales          CDB285DD0DF7AC4DE0400E98195604F3
CDB285DD0E0CAC4DE0400E98195604F3      Sales Assistant    Sales          CDB285DD0DF7AC4DE0400E98195604F3
CDB285DD0E0DAC4DE0400E98195604F3      Manager            Management     CDB285DD0DF9AC4DE0400E98195604F3
CDB285DD0E0EAC4DE0400E98195604F3      Assistant          Management     CDB285DD0DF9AC4DE0400E98195604F3
```

SELECT * FROM customer;

```
PID                                        Balance              IS_PREFERRED_CUSTOMER
--------------------------------------------------------------------------------------------------------------------------------------------
--------------------
CDB285DD0E10AC4DE0400E98195604F3        0              1
CDB285DD0E11AC4DE0400E98195604F3        0              1
CDB285DD0E12AC4DE0400E98195604F3        0              0
CDB285DD0E13AC4DE0400E98195604F3        0              0
CDB285DD0E14AC4DE0400E98195604F3        0              0
CDB285DD0E15AC4DE0400E98195604F3        0              1
CDB285DD0E16AC4DE0400E98195604F3        0              0
CDB285DD0E17AC4DE0400E98195604F3        0              1
CDB285DD0E18AC4DE0400E98195604F3        0              1
```

SELECT * FROM cust_payments;

```
pid                                        cust_id                              payment_amount
payment_date
--------------------------------------------------------------------------------------------------------------------------------------------
--------------------
CDB285DD0E1FAC4DE0400E98195604F3  CDB285DD0E12AC4DE0400E98195604F3   200    04-MAR-12
CDB285DD0E20AC4DE0400E98195604F3  CDB285DD0E12AC4DE0400E98195604F3   200    05-APR-12
CDB285DD0E21AC4DE0400E98195604F3   CDB285DD0E16AC4DE0400E98195604F3   100    06-MAY-12
CDB285DD0E22AC4DE0400E98195604F3   CDB285DD0E16AC4DE0400E98195604F3   20     07-FEB-12
CDB285DD0E23AC4DE0400E98195604F3   CDB285DD0E16AC4DE0400E98195604F3   30     08-MAR-12
CDB285DD0E24AC4DE0400E98195604F3   CDB285DD0E10AC4DE0400E98195604F3   40     09-FEB-12
CDB285DD0E25AC4DE0400E98195604F3   CDB285DD0E10AC4DE0400E98195604F3   50     10-FEB-12
```

## 4) Interactive SQL Queries

### 4.1 Queries for tasks and operations

#### Task 1 INFORMATION PROCESSING

**a) Enter/update/delete basic information about staff**

Query to enter staff information
Precondition: Assuming customer with pid = 1 does not exists in database
**SQL>** INSERT INTO staff VALUES (1, 'Sales Assistant', 'Sales', (SELECT id FROM store WHERE short_name = 'NewYork'));

Query to update staff information
Precondition: Assuming customer with pid = 1 exists in database
**SQL>** UPDATE staff SET job_title = ‘Manager’ WHERE pid = 1;

Query to delete staff information
Precondition: Assuming customer with pid = 1 exists in database
**SQL>** DELETE FROM staff WHERE pid = 1;

**b) Enter/update/delete basic information about customer**

Enter customer information
Precondition: Assuming customer with pid = 9 does not exists in database
**SQL>** INSERT INTO customer VALUES (9, 0, 1);

Update customer information (*updating a customer from normal to preferred customer)*
Precondition: Assuming customer with pid = 9 exists in database
**SQL>** UPDATE customer SET is_preferred_cust = 1 WHERE pid = 9;

Delete customer information
Precondition: Assuming customer with pid = 9 exists in database
**SQL>** DELETE FROM customer WHERE pid = 9;

**c) Enter/update/delete basic information about vendors and contracts**

Using assumption that only information about contracted vendors is stored in vendors table
Query to enter information about vendor
**SQL>** INSERT INTO vendors VALUES (SYS_GUID(), 'Harper Collins', '5126 Bur Oak Cir, Raleigh, NC', '919-666-7777');

Query to update information about a vendor
**SQL>** UPDATE vendors set phone_number='919-666-7778' WHERE name='Harper Collins';

Query to delete information about a vendor
**SQL>** DELETE FROM vendors WHERE name='Harper Collins';

Query to add new vendor order

```
INSERT INTO vendor_order_contents VALUES(
(SELECT id from vendor_order where vendor_id =(SELECT vendor_id from merchandise WHERE
name='Sweet Tooth')),
(SELECT id from merchandise WHERE name='Sweet Tooth'), 1);
```

d) **Enter/update/delete basic information about warehouse**

Query to add new warehouse contents

```
INSERT INTO warehouse_contents VALUES (
(SELECT id from merchandise WHERE name='Sweet Tooth'), 1000, 2, 3);
```

Query to update warehouse contents

```
UPDATE warehouse_contents SET qty_in_stock=qty_in_stock+1 WHERE mer_id=(SELECT id FROM
merchandise WHERE name='Sweet Tooth');
```

Query to delete warehouse contents

```
DELETE FROM warehouse_contents WHERE mer_id=(SELECT id FROM merchandise WHERE name='Sweet
Tooth');
```

d) **Enter/update/delete basic information about staff/customers**

Query to enter new customer order

```
INSERT INTO customer_order VALUES (
SYS_GUID(),
'NOT SHIPPED', (SELECT to_date(to_char(sysdate,'yyyy/mm/dd hh24:mi:ss'),'yyyy/mm/dd hh24:mi:ss')
FROM dual),
(SELECT pid FROM people WHERE name='John' AND phone='919-111-1111'),
(SELECT pid FROM people WHERE name='Jack' AND phone='445-243-9876')
);
```

**e) Check available books and sell what's in stock to each customer according to their request**

Check available books:

```
SELECT m.name, m.author FROM merchandise m JOIN warehouse_contents w ON w.mer_id=m.id;
```

Output:

```
Sweet Tooth
Ian McEwan

Dear Life: Stories
Alice Munro

Dear Life: Stories
Alice Munro

Days of Blood and Starlight
Laini Taylor

Hallucinations
Oliver Sacks
```

Check available books written by Ian McEwan:

```
SELECT m.name FROM merchandise m JOIN warehouse_contents w ON w.mer_id=m.id AND
m.author='Ian McEwan';
```

Output:

```
Sweet Tooth
```

## Task 2 Maintaining purchase usage records for each vendor

### a) Enter/update a new purchase record for each vendor for each billing cycle

Query to enter a new purchase record for a vendor for a billing cycle
Assuming a purchase record for a vendor is entered into the database at the first of each month.
**SQL>** INSERT INTO vendor_order VALUES (SYS_GUID(), 'NOT_SHIPPED', TO_DATE('2012/10/01 08:00:00', 'yyyy/mm/dd hh24:mi:ss'), '9510', (SELECT id from vendors where name='Harper Collins'));

Query to update a new purchase record for a vendor for a billing cycle
Updating the status of the vendor order from NOT_SHIPPED to DELIVERED
**SQL>** UPDATE vendor_order SET status='DELIVERED' WHERE vendor_id=(SELECT id from vendors where name='Harper Collins') AND order_date=TO_DATE('2012/10/01 08:00:00', 'yyyy/mm/dd hh24:mi:ss');

## Maintaining billing accounts

**a) Generate/maintain billing accounts for every billing cycle of every vendor**

Billing account of vendor is stored in vendor_payments table.
As per assumption that the Bookstore pays exactly the same amount in vendor_payment as the cost in vendor_order of the same month

Example 1: Display billing accounts for vendor ‘Harper Collins’ for the year 2012

```
SQL> SELECT vendor_payments.id,
        vendor_payments.payment_date,vendor_payments.amount,
        FROM vendor_payments, vendors
        WHERE vendors.id=vendor_payments.vendor_id
        AND vendors.name='Harper Collins'
        AND vendor_payments.payment_date > TO_DATE('2012/01/01 00:00:00', 'yyyy/mm/dd hh24:mi:ss')
        AND vendor_payments.payment_date < CURRENT_DATE;
```

```
ID                                 PAYMENT_DATE      AMOUNT
--------------------------------   ------------      -----------
CDA07C78655D840DE0400E9819564AA0   01-01-12          1750
CDA07C78655E840DE0400E9819564AA0   01-02-12           350
CDA07C78655F840DE0400E9819564AA0   01-03-12           455
CDA07C786560840DE0400E9819564AA0   01-04-12          1030
CDA07C786561840DE0400E9819564AA0   01-05-12          1210
CDA07C786562840DE0400E9819564AA0   01-06-12          1600
CDA07C786563840DE0400E9819564AA0   01-07-12          1200
CDA07C786564840DE0400E9819564AA0   01-08-12          1705
CDA07C786565840DE0400E9819564AA0   01-09-12           525
CDA07C786566840DE0400E9819564AA0   01-10-12           715
```

Example 2: Generate billing accounts for a vendor ‘Harper Collins’ for the billing cycle of October 201
Using assumption that a vendor payment is made by the bookstore on the first of every month.

```
SQL> INSERT INTO vendor_payments VALUES (SYS_GUID(), TO_DATE('2012/10/01 16:00:00', 'yyyy/mm/dd
hh24:mi:ss'), '715', (SELECT id from vendors where name='Harper Collins'));
```

**b) Generate/maintain billing accounts for every billing cycle of every customer**

Example 1: Given a month generate total pending balance of a customer based on his purchases made during that month.

Assuming that customer with name = ‘Narla’ and phone number = ‘445-243-9880’ made purchase in November 2012 and we want to generate / retrieve pending balance for the month of november for this customer

```
SQL: SELECT SUM(MerchandiseCost) AS Pending Balance FROM (
                    SELECT coc.mer_id, (coc.quantity * wc.selling_price) AS MerchandiseCost
                             FROM customer_order_contents coc, warehouse_contents wc WHERE
                             coc.order_id IN
                                      (SELECT co.id FROM customer_order co
                                               where co.order_date > TO_DATE('2012/11/01 00:00:00', 'yyyy/mm/dd
                                      hh24:mi:ss') AND
                                               co.order_date < TO_DATE('2012/12/01 00:00:00', 'yyyy/mm/dd
                                      hh24:mi:ss'
                                      )
                             AND
                             co.customer_pid = (
                                      SELECT pid FROM people p WHERE
                                      p.Name = 'Narla' AND p.phone = '445-243-9880'
                                               )
                             )
          );
```

Output:

```
--------------------------------------------------------------------------------------------------------------------------------
Pending Balance
--------------------------------------------------------------------------------------------------------------------------------
10000
```

Example 2: Display all payments made by a customer for a specific time period:
Assuming customer with name = Lucy and phone = ‘445-243-9878’ made couple of payments and we’re trying to retrieve all payments made during the duration 1-Feb-12 to 30-May-12

```
SQL SELECT cp.id as PaymentID, cp.payment_amount, cp.payment_date FROM
                   cust_payments cp WHERE
                             cp.cust_id IN
                             (SELECT pid FROM people p WHERE
                                      p.Name = 'Lucy' AND p.phone = '445-243-9878'
                             ) AND
                             cp.payment_date > TO_DATE('2012/02/01 00:00:00', 'yyyy/mm/dd hh24:mi:ss') AND
                             cp.payment_date  < TO_DATE('2012/05/30 00:00:00', 'yyyy/mm/dd hh24:mi:ss')
                             ;
```

Output:

```
------------------------------------------------------------------------------------------------------------------------------------------
PaymentID                                              payment_amount         payment_date
------------------------------------------------------------------------------------------------------------------------------------------
CDB285DD0E61AC4DE0400E98195604F3     100            06-MAY-12
CDB285DD0E62AC4DE0400E98195604F3     20             07-FEB-12
CDB285DD0E63AC4DE0400E98195604F3     30             08-MAR-12
```

Example 3: Total amount of payment made by a customer for a given time period

```
SQL SELECT sum(cp.payment_amount) AS TotalPaymentAmountFROM
                    cust_payments cp WHERE
                              cp.cust_id IN
                              (SELECT pid FROM people p WHERE
                                        p.Name = 'Lucy' AND p.phone = '445-243-9878'
                              ) AND
                              cp.payment_date > TO_DATE('2012/02/01 00:00:00', 'yyyy/mm/dd hh24:mi:ss') AND
                              cp.payment_date  < TO_DATE('2012/05/30 00:00:00', 'yyyy/mm/dd hh24:mi:ss')
                              ;
```

Output:

```
------------------------------------------------------------------------------------------------------------------------------------------
TotalPaymentAmount
------------------------------------------------------------------------------------------------------------------------------------------
150
```

## Task 4: Reports

**a) Report the purchase history for a given customer and for a certain time period (day/month/year)**

Assuming that customer with name = ‘Narla’ and Phone no = ‘445-243-9880' made purchase in November 2012 and we want to retrieve his purchase history for the same month. By purchase history we mean merchandises name, author, book info, quantity, date of order which are purchased by a particular customer during the given time period.

```
Query: SELECT coc.order_id AS orderID, wc.mer_id as merchandiseID, mer.name, mer.author,
            mer.book_info, coc.quantity, co.order_date FROM
                customer_order_contents coc, warehouse_contents wc,
                merchandise mer, customer_order co WHERE
                    coc.mer_id = wc.mer_id AND
                    wc.mer_id = mer.id AND
                    coc.order_id = co.id AND
                    co.customer_pid IN (
                        SELECT pid FROM people p WHERE
                        p.Name = 'Narla' AND p.phone = '445-243-9880'
                            ) AND
                    co.order_date > TO_DATE('2012/11/01 00:00:00', 'yyyy/mm/dd hh24:mi:ss')
                AND co.order_date < TO_DATE('2012/12/01 00:00:00', 'yyyy/mm/dd hh24:mi:ss');
            order by coc.order_id;
```

Output:

```
-----------------------------------------------------------------------------------------------------------------------------------------------
-----------------
orderID                                          merchandisID                                 name --
-- author                      book_info         Quantity                 order_date
-----------------------------------------------------------------------------------------------------------------------------------------------
-----------------
CDB285DD0ED8AC4DE0400E98195604F3   CDB285DD0EB7AC4DE0400E98195604F3 Hallucinations
Oliver Sacks                   Fiction           1                        04-NOV-12
```

**b) Return information on all the vendors a particular store has contract with**

Since we store only the contracted vendors in vendors table, query to return information about vendors.

**SQL>** SELECT * FROM vendors;

```
ID                                                    NAME                    ADDRESS
PHONE_NUMBER
---------------------------------------------   -----------------------   ------------------------------------------   ---------------------
CDA04D6ABAFB1990E0400E98195649FB    Harper Collins          5126 Bur Oak Cir, Raleigh, NC              919-666-
7777
CDA04D6ABAFC1990E0400E98195649FB    Schiel and Denver     5122 Bur Oak Cir, Raleigh, NC              919-555-
6666
CDA04D6ABAFD1990E0400E98195649FB   Random House         3101 Hillsborough St, Raleigh, NC        919-444-
5555
CDA04D6ABAFE1990E0400E98195649FB    Ivy House                3618 Cannold Ct, Raleigh, NC              919-111-
2222
```

**c) Find all vendor orders which haven't been shipped yet**

**SQL>** SELECT * FROM vendor_order WHERE status='NOT SHIPPED';

**Return information on all the customers a given salesperson assisted during a certain time period (day/month/year)**

<u>Return information on all the customers the salesperson 'Jim' assisted during the time period 2012/05/01 to 2012/12/01</u>

```
SELECT name, gender, address,phone from people WHERE pid=
(SELECT co.customer_pid FROM customer_order co WHERE co.sales_person_pid=(SELECT pid from people WHERE name='Jim' AND phone='444-222-3333') AND co.order_date > TO_DATE('2012/05/01 00:00:00', 'yyyy/mm/dd hh24:mi:ss') AND co.order_date < TO_DATE('2012/12/1 00:00:00', 'yyyy/mm/dd hh24:mi:ss'));
```

Output:

```
        Narla
        Female
        2512, Avery Close, Raleigh, NC
        445-243-9876
```

**d) Return information on Books-A-Thousand' staff grouped by their role.**

Precondition: Assuming there exists staff enteries in staff table.

Query: SELECT s.dept_name, s.job_title, p.name, p.ssn, p.gender, p.address, p.phone, p.date_of_birth, p.pid, s.store_id
FROM people p JOIN staff s ON p.pid = s.pid ORDER BY s.dept_name;

```
Dept_name    Job_title       name   ssn     gender      address                    phone          date_of_birth
PID                                               Store_id
-----------------------------------------------------------------------------------------------------------------------------------
Management     Manager         Andy  1253432 Male MOUNT ABU, Rajesthan, India 555-111-5555        01-JAN-84
CDB285DD0E0DAC4DE0400E98195604F3           CDB285DD0DF9AC4DE0400E98195604F3
Management    Assistant         Jim    1234789 Male 1621 Times Square, Newyork 444-222-3333         01-JAN-83
CDB285DD0E0EAC4DE0400E98195604F3           CDB285DD0DF9AC4DE0400E98195604F3
Sales              Cashier             Joseph 1234589 Male   123 HILLS, RALEIGH, NC 919-111-2222           01-JAN-81
CDB285DD0E0BAC4DE0400E98195604F3           CDB285DD0DF7AC4DE0400E98195604F3
Sales              SalesAssistant    John 1234567    Male      123 CREST RD, RALEIGH, NC      919-111-1111              01-
JAN-85 CDB285DD0E0AAC4DE0400E98195604F3            CDB285DD0DF7AC4DE0400E98195604F3
Sales              SalesAssistant    Michael 1231234  Male 123 AVENT FERRY RD, RALEIGH, NC        222-333-1111
01-JAN-87 CDB285DD0E0CAC4DE0400E98195604F3        CDB285DD0DF7AC4DE0400E98195604F3
```

**4.2 Autotrace and Indexes for two tables**

1) The first index is added to vendor_payments based on the "payment_date" column

```
SQL> SET AUTOTRACE ON
SQL> SELECT * FROM vendor_payments, vendors
 WHERE vendors.id=vendor_payments.vendor_id
 AND vendors.name='Harper Collins'
 AND vendor_payments.payment_date > TO_DATE('2012/01/01 00:00:00', 'yyyy/mm/dd hh24:mi:ss')
 AND vendor_payments.payment_date < CURRENT_DATE;

10 rows selected.

Execution Plan
----------------------------------------------------------
   0      SELECT STATEMENT Optimizer=ALL_ROWS (Cost=4 Card=11 Bytes=18
          04)
   1    0   FILTER
   2    1   NESTED LOOPS (Cost=4 Card=11 Bytes=1804)
   3    2            TABLE ACCESS (BY INDEX ROWID) OF 'VENDORS' (TABLE) (Co
          st=1 Card=1 Bytes=122)
   4    3            INDEX (UNIQUE SCAN) OF 'SYS_C0044031' (INDEX (UNIQUE
          )) (Cost=1 Card=1)
   5    2            TABLE ACCESS (FULL) OF 'VENDOR_PAYMENTS' (TABLE) (Cost
          =3 Card=11 Bytes=462)


Statistics
----------------------------------------------------------
          0  recursive calls
          0  db block gets
          10  consistent gets
          0  physical reads
          0  redo size
          1638  bytes sent via SQL*Net to client
          435  bytes received via SQL*Net from client
          2  SQL*Net roundtrips to/from client
          0  sorts (memory)
          0  sorts (disk)
          10  rows processed
```

**SQL>** CREATE INDEX vendor_payments_date_index ON vendor_payments(payment_date);

After creating index on “payment_date”

```
SQL> SELECT * FROM vendor_payments, vendors
 WHERE vendors.id=vendor_payments.vendor_id
 AND vendors.name='Harper Collins'
 AND vendor_payments.payment_date > TO_DATE('2012/01/01 00:00:00', 'yyyy/mm/dd hh24:mi:ss')
 AND vendor_payments.payment_date < CURRENT_DATE;

10 rows selected.

Execution Plan
----------------------------------------------------------
   0      SELECT STATEMENT Optimizer=ALL_ROWS (Cost=3 Card=11 Bytes=18
          04)

   1    0   FILTER
   2    1   NESTED LOOPS (Cost=3 Card=11 Bytes=1804)
   3    2           TABLE ACCESS (BY INDEX ROWID) OF 'VENDORS' (TABLE) (Co
          st=1 Card=1 Bytes=122)

   4    3            INDEX (UNIQUE SCAN) OF 'SYS_C0044031' (INDEX (UNIQUE
          )) (Cost=1 Card=1)

   5    2           TABLE ACCESS (BY INDEX ROWID) OF 'VENDOR_PAYMENTS' (TA
          BLE) (Cost=2 Card=11 Bytes=462)

   6    5           INDEX (RANGE SCAN) OF 'VENDOR_PAYMENTS_DATE_INDEX' (
          INDEX) (Cost=1 Card=13)


Statistics
----------------------------------------------------------
          9  recursive calls
          0  db block gets
          16  consistent gets
          0  physical reads
          0  redo size
```

```
     1638  bytes sent via SQL*Net to client
     435  bytes received via SQL*Net from client
     2  SQL*Net roundtrips to/from client
     0  sorts (memory)
     0  sorts (disk)
     10  rows processed
```

Query:

```
SELECT name, gender, address,phone from people WHERE pid=
(SELECT co.customer_pid FROM customer_order co WHERE co.sales_person_pid=(SELECT pid from people WHERE
name='Jim' AND phone='444-222-3333') AND co.order_date > TO_DATE('2012/05/01 00:00:00', 'yyyy/mm/dd
hh24:mi:ss') AND co.order_date < TO_DATE('2012/12/1 00:00:00', 'yyyy/mm/dd hh24:mi:ss'));
```

Output:

```
Narla
Female
2512, Avery Close, Raleigh, NC
445-243-9876
```

```
Execution Plan
----------------------------------------------------------
   0      SELECT STATEMENT Optimizer=ALL_ROWS (Cost=6 Card=1 Bytes=73)
   1    0   TABLE ACCESS (BY INDEX ROWID) OF 'PEOPLE' (TABLE) (Cost=1
          Card=1 Bytes=73)

   2    1     INDEX (UNIQUE SCAN) OF 'SYS_C0064013' (INDEX (UNIQUE)) (
          Cost=0 Card=1)

   3    2       TABLE ACCESS (FULL) OF 'CUSTOMER_ORDER' (TABLE) (Cost=
          3 Card=1 Bytes=42)

   4    3         TABLE ACCESS (BY INDEX ROWID) OF 'PEOPLE' (TABLE) (C
          ost=2 Card=1 Bytes=23)

   5    4           INDEX (RANGE SCAN) OF 'SYS_C0064015' (INDEX (UNIQU
          E)) (Cost=1 Card=1)
```

```
Statistics
----------------------------------------------------------
    421  recursive calls
      0  db block gets
    146  consistent gets
      0  physical reads
      0  redo size
    556  bytes sent via SQL*Net to client
    435  bytes received via SQL*Net from client
      2  SQL*Net roundtrips to/from client
      8  sorts (memory)
      0  sorts (disk)
      1  rows processed
```

**<u>Creating Index on customer_pid</u>**
**SQL>  CREATE INDEX customer_order_idx ON customer_order(sales_person_pid);**

```
Index created.

SQL> SELECT name, gender, address,phone from people WHERE pid=
(SELECT co.customer_pid FROM customer_order co WHERE co.sales_person_pid=(SELECT pid from people WHERE name='Jim' AND phone='444-222-3333') AND co.order_date > TO_DATE('2012/05/01 00:00:00', 'yyyy/mm/dd hh24:mi:ss') AND co.order_date < TO_DATE('2012/12/1 00:00:00', 'yyyy/mm/dd hh24:mi:ss'));  2
Narla
Female
2512, Avery Close, Raleigh, NC
445-243-9876



Execution Plan
----------------------------------------------------------
   0      SELECT STATEMENT Optimizer=ALL_ROWS (Cost=5 Card=1 Bytes=73)
   1    0   TABLE ACCESS (BY INDEX ROWID) OF 'PEOPLE' (TABLE) (Cost=1
          Card=1 Bytes=73)

   2    1     INDEX (UNIQUE SCAN) OF 'SYS_C0064013' (INDEX (UNIQUE)) (
          Cost=0 Card=1)
```

**3    2       TABLE ACCESS (BY INDEX ROWID) OF 'CUSTOMER_ORDER' (TAB**

```
        LE) (Cost=2 Card=1 Bytes=42)

   4    3         INDEX (RANGE SCAN) OF 'CUSTOMER_ORDER_IDX' (INDEX) (
        Cost=1 Card=1)

   5    4           TABLE ACCESS (BY INDEX ROWID) OF 'PEOPLE' (TABLE)
        (Cost=2 Card=1 Bytes=23)

   6    5             INDEX (RANGE SCAN) OF 'SYS_C0064015' (INDEX (UNI
        QUE)) (Cost=1 Card=1)


Statistics
----------------------------------------------------------
          1  recursive calls
          0  db block gets
          6  consistent gets
          0  physical reads
          0  redo size
        556  bytes sent via SQL*Net to client
        435  bytes received via SQL*Net from client
          2  SQL*Net roundtrips to/from client
          0  sorts (memory)
          0  sorts (disk)
          1  rows processed
```

## 4.3)Query Correctness Proof:

**1) To list the author and merchandize name that for the items that exist in the warehouse.**

SELECT m.name, m.author FROM merchandise m JOIN warehouse_contents w ON w.mer_id=m.id;

R1 = Merchandise JOIN $_{\text{(merchandize.mer_id=warehouse_contents.id)}}$Warehouse_contents
R2 = Project $_{\text{(merchandize.name,merchandize.author)}}$ R1

The relational algebra join the warehouse_contents table with the merchandise table agreeing on the same id of the merchandize table with mer_id of the warehouse_contents table. That is the we now have records that are present both the tables. From this we project out the author name and name of the merchandize. Thus this will list out those merchandize that are present in the warehouse table.

**2) Return information on all the customers the salesperson 'Jim' assisted during the time period 2012/05/01 to 2012/12/01**

```
SELECT name, gender, address,phone from people WHERE pid= (SELECT co.customer_pid FROM customer_order
co WHERE co.sales_person_pid=(SELECT pid from people WHERE name='Jim') AND co.order_date >
TO_DATE('2012/05/01 00:00:00', 'yyyy/mm/dd hh24:mi:ss') AND co.order_date < TO_DATE('2012/12/1
00:00:00', 'yyyy/mm/dd hh24:mi:ss'));
```

R1= Select $_{\text{name='John'}}$ People
R2= Project $_{\text{pid}}$ R1
R3= Select $_{\text{sales\_person\_pid=R2 AND order\_date>'2012/05/01' AND order\_date<'2012/12/1'}}$ Customer_order
R4= Project $_{\text{customer\_order.customer\_pid}}$ R3

For the query above we first select the records that have the name as 'John' (given name) and we project out the pid (*pid'*) of this salesperson. Using this pid we select records from Customer_order table having same salesperson id . This will list out all the customers that have been assisted by the given salesperson.On applying the further filtering based on date (order_date>'2012/05/01' AND order_date<'2012/12/1'). This will list out only those customers that have been assisted by the given salesperson in between the given dates. Hence the query gives the required output as is thus correct.