IST 302 Database Project
Database Project – Instructions

This semester's project involves the design of a database system for a Sales Transaction application. In these instructions we refer to the data that would result from execution of a business query (BQ) as a Business View (BV). Note that a BV could be a virtual view or a materialized view.

Each Project Team will be required to do the following:

A. Business Understanding 1 (Be prepared to discuss this on 11/18/2015)

1. Do some background reading on *Sales Transactions* applications

A sale is an exchange of a product (or service) between two or more parties in exchange of money or some other compensation. It takes place almost everyday in human-life. Whenever there is a sale, there are at least two parties involved; a seller who is typically an owner of the product or who has legal possession and a buyer who is seeking the product.

Transaction Detail—this table typically contains one row per item in a transaction basket. The transaction detail table is often very wide, containing dozens of attributes about each item in a transaction. Captured fields typically include the following:

- Pricing information such as retail price, discounted price and markdown amounts;
- Manually keyed price changes at the point of sale;
- UPC codes, which can tie to vendors;
- Package sizes and weights;
- Timestamps for each item; and
- Tax amounts (sometimes) by item.

2. Identify & describe typical types of users (at least 3) who would need transactional information based on Sales Transactions;

B. Data Understanding (Be prepared to discuss this on 11/18/2015)

3. Familiarize yourself with each of the operational database tables of the LIY26 schema by exploring the data in the tables (see Appendix 1):
    - Determine if the datatype that is used in the columns definition is consistent with the actual data. Provide justification
    - Identify columns which appear to contain useless data. Provide justification.
    - Analyze each low cardinality field to determine the set of values; for each field.
    - Identify attributes that appear to be potential identifiers of embedded entities. Provide justification
    - Identify functional dependencies. Provide justification
    - Identify tables that appear to represent the same entity. Provide justification

C. Conceptual Data Modeling (Be prepared to discuss this on 11/18/2015)

4. Given the results of your Data Understanding activities, create an appropriate ERD for your *Sales Transactions* database system. Provide justification

D. Business Understanding 2 (Be prepared to discuss this on 12/02/2015)

5. For each user, identify & describe at least one transactional information needs (i.e. *Business Query*).
6. Data from each table must be included in at least one *Business Query*. See Appendix 3.

7. Some of your transactional questions should involve the following derived attributes:
    - Discount = List_Price - Actual_Price, where Actual_Price = Amount_Sold/Quantity_Sold
    - Discount_Rate = Discount*100/List_Price
    - Days_To_Ship = Shipment_Date – Sale_Date
    - Days_To_Pay = Payment_Date – Sale_Date

E. Security Requirements (Be prepared to discuss this on 12/02/2015):

8. Based on the query profile that you developed, place an "X" in each cell of Table E1 for which the relevant BUSINESS VIEW should be available to the given USER

Table E1: INITIAL ACCESS PERMISSION INTERACTION MATRIX

	BUSINESS VIEWs						
USERs	BV01	BV02	BV03.			BV…	
U01							
U02							
U03							
U…							

9. Place an "X" in each cell of Table E2 for which the relevant BUSINESS VIEW should not be available to the given USER

Table E2: ACCESS PREVENTION INTERACTION MATRIX

	BUSINESS VIEWs						
USERs	BV01	BV02	BV03. .			BV…	
U01							
U02							
U03							
U..							

F. DESIGN RELATIONAL DATABASE (Be prepared to discuss this on 12/02/2015):

10. Define tables of relational database system that corresponds to your ERD. Each relation should be in 3NF. Provide justification.

D. INTEGRITY CONSTRAINTS (Be prepared to discuss this on 12/02/2015):

You will need to specify & test integrity constraints of your relational database system

11. Explore the values of low cardinality columns in order to get an idea as to the acceptable values for the column
12. Define domain constraints for each low cardinality column.

13. For each base table:
    - define primary key constraint;
    - Where appropriate, define referential integrity constraints
    - Where appropriate, define relevant inter-columns.

E. CREATE ORACLE TABLES & LOAD DATA into ORACLE:

For each table of your relational database system:

14. Estimate the storage requirements
15. Determine the values of the PCTFREE, PCTUSED, INITRANS, PCTINCREASE, etc. parameters
16. Specify the SQL DDL statement to Create the table. Your CREATE statement should include appropriate Integrity Constraint Specifications & appropriate values for the STORAGE clause, PCTFREE, PCTUSED, INITRANS, etc. parameters.
17. Execute the SQL DDL statement to Create the table.
18. Use the data in the corresponding tables of the LIY26 schema to populate the corresponding permanent table permanent tables of your relational database system.
19. Record any integrity constraint violation.

F. DETERMINE ACCESS STRUCTURES FOR DATABASE SYSTEM:

20. For each Business Query in your Query Profile, use the EXPLAIN PLAN facility to generate various execution plans, and identify its 'best' execution plan.
21. Based on the information obtained from these 'best' execution plans, determine the *Indexes* that should be included in your database and those which should be excluded. Provide justification for each choice.
22. Determine which of the *Materialized VIEWS* should be included in your database.

G. Specify & Test Security Scripts:

23. Write SQL Scripts to implement the security requirements.

24. Do Positive & Negative Tests on the SQL Security Scripts.

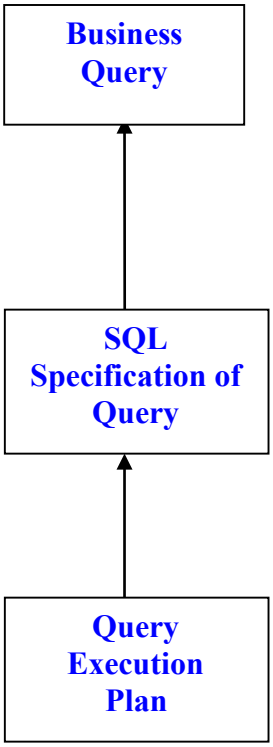
H. Team Contribution

25. Each student should write a paragraph about their team members contributions (e.g., what percentage of contribution a team member has and in what area). The team contribution report is sent to instructor separately as an individual assignment on Canvas.

Appendix 1: Operational Database Tables

TableName	Description	Columns
LI_CHANNELS	Data on Distribution Channels	**CHANNEL_ID**, CHANNEL_DESC, CHANNEL_CLASS, CHANNEL_TOTAL
LI_CUSTOMERS_INTX	Data on Customers obtained from internal source	**CUST_ID**, CUST_FIRST_NAME, CUST_LAST_NAME, CUST_GENDER ,CUST_MAIN_PHONE_NUMBER ,CUST_EMAIL, CUST_STREET_ADDRESS ,CUST_POSTAL_CODE ,CUST_CITY ,CUST_STATE_PROVINCE ,C.COUNTRY_ID ,CUST_TOTAL, COUNTRY_NAME ,COUNTRY_SUBREGION ,COUNTRY_REGION, COUNTRY_TOTAL
LI_CUSTOMERS_EXT	Data on Customers obtained from external sources	**CUST_ID**, CUST_GENDER, CUST_YEAR_OF_BIRTH, CUST_MARITAL_STATUS, CUST_INCOME_LEVEL, CUST_CREDIT_LIMIT
LI_PRODUCTS	Product data	**PROD_ID**, PROD_NAME, PROD_DESC, PROD_SUBCATEGORY, PROD_SUBCAT_DESC, PROD_CATEGORY, PROD_CAT_DESC, PROD_WEIGHT_CLASS, PROD_UNIT_OF_MEASURE, PROD_PACK_SIZE, SUPPLIER_ID, PROD_STATUS, PROD_LIST_PRICE, PROD_MIN_PRICE, PROD_TOTAL
LI_PROMOTIONS	Promotions data	**PROMO_ID**, PROMO_NAME, PROMO_SUBCATEGORY, PROMO_CATEGORY, PROMO_COST, PROMO_BEGIN_DATE, PROMO_END_DATE, PROMO_TOTAL
LI_SALES_12_13	Sales transaction data for 2012 through 2013	**SALESTRANS_ID,** *PROD_ID, CUST_ID, CHANNEL_ID, PROMO_ID, SALE_DATE*, SHIPPING_DATE, PAYMENT_DATE, QUANTITY_SOLD, AMOUNT_SOLD, UNIT_PRICE
LI_SALES_14	Sales transaction data for 2014	**SALESTRANS_ID,** *PROD_ID, CUST_ID, CHANNEL_ID, PROMO_ID, SALE_DATE*, SHIPPING_DATE, PAYMENT_DATE, QUANTITY_SOLD, AMOUNT_SOLD, UNIT_PRICE

APPENDIX 2: Business Query, SQL Specification, Query Execution Plan



Appendix 3: Guidelines for Business Queries

Each Business Query (*BQ*) may be represented by one or more SQL Queries (SQs). In developing the set of *BQ*s that constitute your Business Query Profile, identify *BQ*s that would result in *SQ*s having the following characteristics:

1. **Detail Queries: Single Table**
    a. Involving a column for which a B-Tree Index would be relevant
    b. Involving a column for which a Bitmap Index would be relevant

    You should experiment with INTERSECT, MINUS operators where possible

2. **Detail Queries: Two Tables**
    c. EQUI JOIN without or with σ condition
    d. Right or left join with complex σ condition

3. **Detail Queries: Three Tables**
    Same as for Two Tables with at least one being either LI_SALES_98_99 or KM_SALES_2K.

4. **Detail Queries: Four Tables**
    Same as for Two Tables with at least one table being either KM_SALES_98_99 or KM_SALES_2K, and another being KM_COUNTRIES.

5. **Summary Queries: Four Tables**
    Same as Detail Queries with Four Tables but involving SUM, AVG, COUNT, MAX, MIN operators.

6. **Summary Queries: Four Tables using** Materialized Views
    Create Materialized Views that could be used to provide the same output as the Summary Queries above. Calculate the storage requirements for each materialized view and include an appropriate STORAGE clause when you create the table. Run each summary query against the relevant Materialized View, and in each case determine if there is a difference in the estimated execution cost as compared to when the query is run against the base tables.