DATABASE MANAGEMENT SYSTEM(CS310)

GROUP PROJECT

VEHICLE INSURANCE COMPANY

~UNDER THE GUIDANCE OF DR UMA SHESHADRI

& PRAMOD YELMELEWAD

TEAM MEMBERS

Brij Vaghani	20BCS033
Daulat Jha	20BCS037
Hemant Dhawale	20BCS058
Hitarth Vyas	20BCS060
Keshav Jha	20BCS070
Lucky Yadav	20BCS077
Rohan Singh	20BCS113
Tanzeem	20BCS132
Tarun Sharma	20BCS133
Vinayak Upadhyay	20BCS136

ACKNOWLEDGEMENT

We would like to acknowledge everybody responsible for completion of our project. It gives us great pleasure to thank the assistance and contribution of many individuals who have been actively involved at various stages of this project to make it success.

Firstly we are very grateful to or esteemed institute **Indian Institute of Information Technology, Dharwad,** for providing us opportunity to degree the course.

Perfection is impossible to achieve, but we tried our level best to fulfil our goal. We could not have achieved our goal without able guidance of **Dr. Uma Seshadri,** HOD, CS Department; **Dr. Pramod Yelmewad** and **Dr. Supriya Nadiger.** We are indebted to these lecturers, for providing us an opportunity to work under the their guidance and providing their constant support and encouragement.

Finally we convey or acknowledgement to our friends and family who directly or indirectly associated with us in completion of this project.

We Thank one and all.

PART-A
CONCEPTUAL DATA MODEL

INTRODUCTION

The conceptual data model is a structured business view of the data required to support business processes, record business events, and track related performance measures. This model focuses on identifying the data used in the business but not its processing flow or physical characteristics.

DESIGN RULES

To design our car insurance database conceptual data model first we need to decide what characteristics underpin the model under investigation. As a group we decided on various rules that need to be implemented in order for the model to be consistent and precise.
Table 1 below illustrates these rules.

Design Rule	Description	Example
Rule 1	All the individual entity types must be in capital letters.	CUSTOMER
Rule 2	An underscore is used to label an entity type with more than one word.	TERMS_CONDITIONS
Rule 3	Plurals are not used when labelling entity types.	APPLICATION
Rule 4	No abbreviations are used when labelling entity types.	QUOTE
Rule 5	Entity types must not be in Numerical.	VEHICLE
Rule 6	Every entity must contain a Primary key.	Like CUSTOMER_ID for the customer Entity.

Table 1: Design rules for CDM model of car insurance database

ASSUMPTIONS

To design the Conceptual Data Model (CDM) we have a certain set of assumptions. These assumptions will help shape our model to allow consistency within our design. Table 2 presents the assumptions used in this model.

Table 2: Assumptions used in car insurance database model

Assumption	Description
Assumption 1	Customers must have original proof Id's like Aadhar, driving license etc..
Assumption 2	Customer not having personal accidental cases before in on-road
Assumption 3	The online insurance is given to customers over 18 years of age.
Assumption 4	The online insurance needs some driving history of the customer.
Assumption 5	The online insurance needs to know the type of car the customer drives.
Assumption 6	The online insurance needs to know about the insurance history of the customer.

Entity Types:

All of the entity types that we feel are relevant in our CDM are illustrated in Table 3, below.

Table 3: Entity types used in car insurance database system CDM model.

Entity Type	Description
CUSTOMER	Records all the personal details about the customer.
APPLICATION	Records details of the insurance coverage requested by Customer
QUOTE	Records details of customer potential cost of the insurance product.
INSURANCE_POLICY_COVERAGE	Records details of the insurance agreement.
PREMIUM_PAYMENT	Records details of customer cost and payments.
CLAIM	Records details of customer claims in case of an accident.
CLAIM_SETTLEMENT	Records details of settlement made on claims.
STAFF	Records details of employees.
DEPARTMENT	Records details of the various departments.

OFFICE	Records details of different office locations.
MEMBERSHIP	Records details of customer membership,clubs and societies.
VEHICLE_SERVICE	Records details of different car services offered.
NOK	Records details of the next o kin.
COVERAGE	Records all terms and conditions in regard to the policy
INSURANCE_POLICY	Records details of Insurance agreement.
PRODUCT	Records details of the products offered by insurance company
RECEIPT	Details of premium payments to customer
INSURANCE_COMPANY	Details of the insurance organization giving the insurance cover.
VEHICLE	Records details of Vehicle model, cost and registration.
INCIDENT	Records details of the accident, theft, fire, etc.
POLICY_RENEWABLE	Records details of due date of insurance policy.
INCIDENT_REPORT	Records details of the individual incident

Relationships in CDM:

Applying Relationships to Entities:

To apply relationships to our entity types we formed certain assumptions to simplify and determine connections between entity types. These assumptions and explanations are illustrated in Table 4 below.

Table 4: CDM relationship of entities for car insurance database:

Entity Type	Related To Entities	Relationship
QUOTE	APPLICATION	One to one
APPLICATION	INSURANCE POLICY CUSTOMER	One to many One to many
CUSTOMER	MEMBERSHIP PREMIUM_PAYMENT CLAIM VEHICLE	Many to many One to many One to many One to one, one to many
INSURANCE_POLICY	DEPARTMENT NOK	One to many One to many
PREMIUM_PAYMENT	RECEIPT	One to many
CLAIM	CLAIM_SETTLEMENT	One to one
VEHICLE INSURANCE DEPARTMENT	DEPARTMENT SERVICE	One to one, one to many One to many
DEPARTMENT	OFFICE INSURANCE_COMPANY	Many to many One to many
INSURANCE_COMPANY	STAFF	Many to many

Graphical presentation of CDM:

The Conceptual Data Model that will be used as a starting point in designing our online car insurance database system can be seen in Figure 3 (with no entities relationships) and Figure 4 (with entities relationships), done in ERwin software.

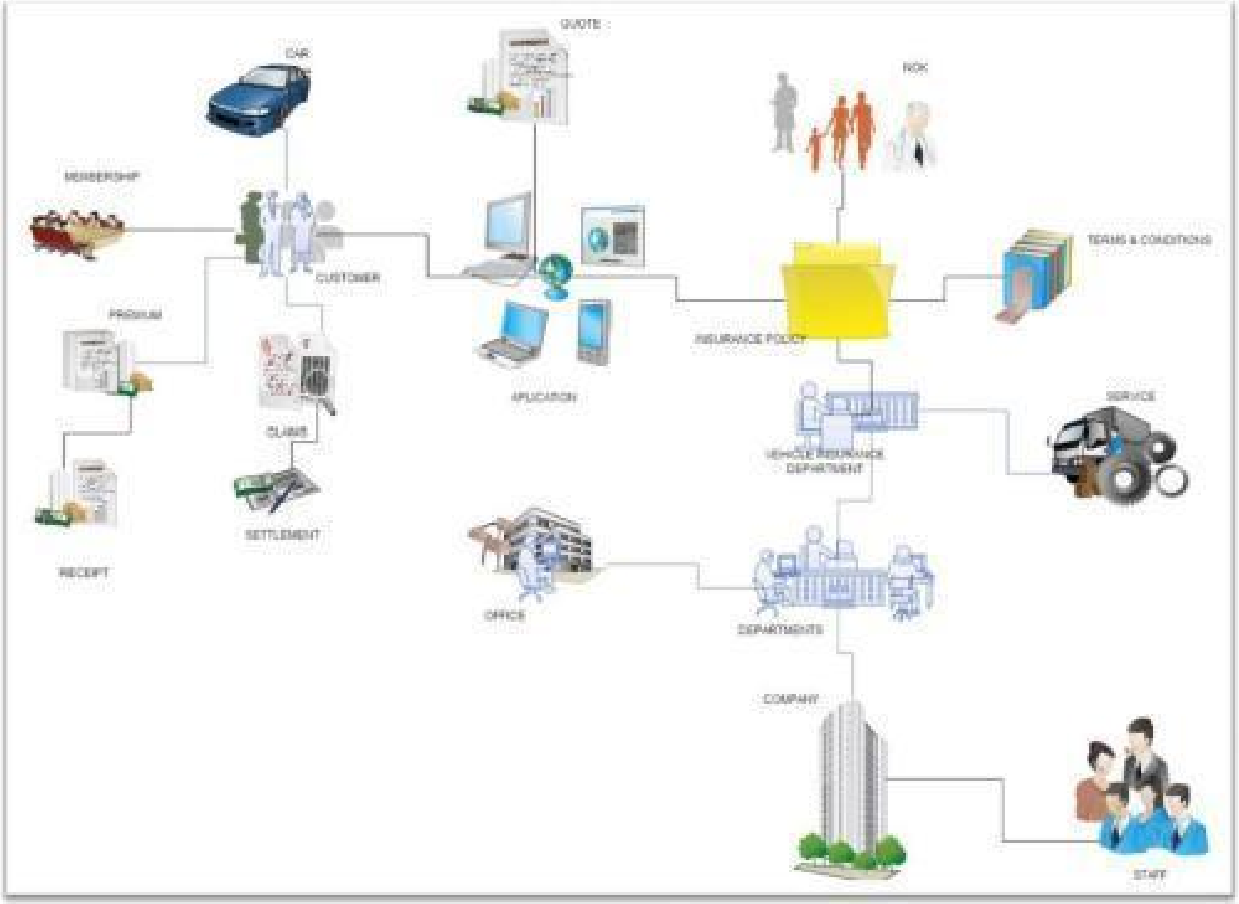
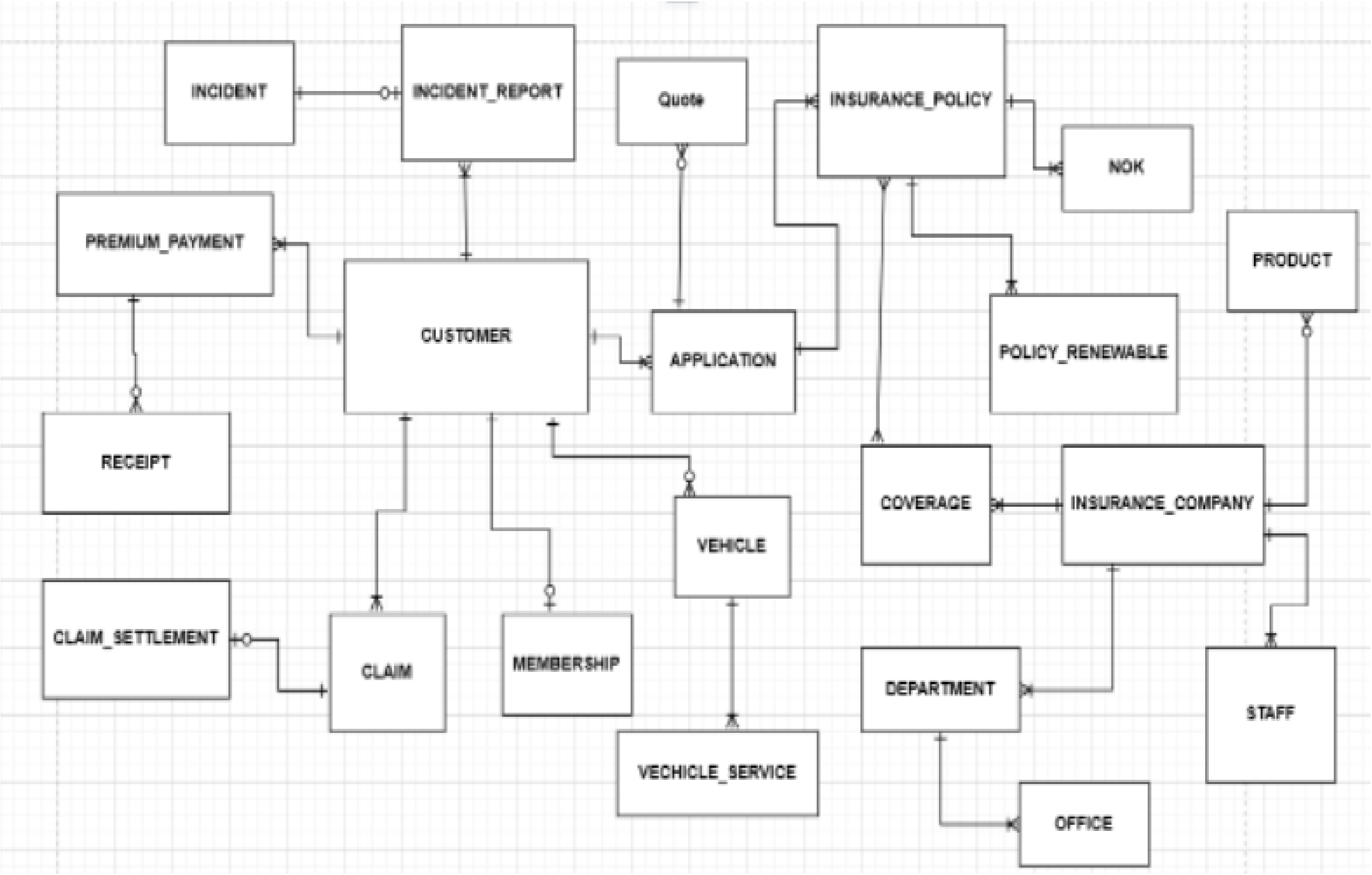


Figure 4: Conceptual data model of Car insurance (with entity types):



PART-B: -LOGICAL DATA MODEL

INTRODUCTION

Part B of this report includes the design of a logical data model (LDM) for vehicle insurance company XYZ Ltd. First, we had to make some changes and improvements to the conceptual data model which are explained and presented in the last part of this report. Then we identify all the attributes in old and new entities and assign them the primary (PK) and foreign keys (FK) and make relationships with them in Erwin to make a full LDM. We identified each variable and the data types that this LDM could be used to design the database in Access or Oracle database management systems.

Introduction of terms used for constructing LDM

Elements

Elements in the data model are named entities. This is any distinguishable object that presents part of the database. It can be related to any object in the real world such as a car, a customer (person), a policy, a company, etc. with respective attributes that are relevant to the software system.

Properties of entities can have values:

- Name
- Description of the meaning and significance.
- Weather entity is dependent or non-dependent.
- List of attributes (Car entity: year, manufacturer, model, mileage, owner, license, book of maintenance) with properties (data type, size, is it required or not).
- The attributes (or attributes) are used to precisely identify an entity (primary key – PK, foreigner key – FK, ...).
- Constraints of individual or combined attributes values (e.g., date of issue of new policy can't be before the renewal date of the policy).
- Rules to grant permission to users or user groups to access the entity.
- Expected number of entity instances and expected growth rate.

Or additional:
- List of attributes to be indexed to optimize access time.
- List of attributes to be encrypted or compressed.
- The weather entity should become a database view or a table.
- Weather entity should become a materialized view.
- List of database triggers to be implemented for that entity.

Relations

Relationship- Designates logical association between entities, with cardinality of the participant

Entities: one-to-one, one-to-many, or many-to-many relationships. Relationships can be identifying or non-identifying (identifying A-B; existence of B depends on existence of A).

Generalization/specialization– Indicates an “is a” relationship between entities. For example, department entity is a generalization of different types of departments; at the same time vehicle Insurance department or travel insurance department is specialization of department entity.

Aggregation- is an abstraction that turns relationship between entities into an aggregate entity, rarely used. Example: “customer-insurance advisor - date” can be an aggregate entity called Appointment.

Constraints

The database normalization technique is used to impose restrictions on data model that is based on dependencies between entities and their attributes. Normalization is used with the goal objective to avoid duplication of information in order to safe guard the consistency (integrity) of the data.

Data types

When we assign attributes to entities with primary keys and foreign keys do the normalization, we identify each attribute with data type for each data management system – Access as seen in example Table 6, below.

Table 6: Example of data types in Access and DMS

Entity type	Attributes	Data type
CUSTOMER	FIRST_NAME	Varchar (10)
	LAST_NAME	Varchar (10)
	DATE_OF_BIRTH	Date

Business Rules, Attributes, Data types and Primary/Foreign Keys

This section of the report identifies all of the attributes, data types and primary and foreign keys for our system LDM. For better overview we present the table number to the corresponding entity type, followed by the business rule of what we wish the entity type to capture, posted below in Table 7.

Table 7: Table number entity type

Table Number	Entity Type	Business Rules
1	CUSTOMER	Records all the personal details about the customer
2	APPLICATION	Records details of the insurance cover requested by the customer.
3	QUOTE	Records details of customer potential cost of the insurance product.
4	INSURANCE_POLICY	Records details of Insurance agreement.

5	PREMIUM_PAYMENT	Records details of customer cost of payments.
6	VEHICLE	Records details of Vehicle model, cost and registration.
7	CLAIM	Records details of customer claims in case of an incident.
8	CLAIM SETTLEMENT	Records details of settlement made on claims
9	STAFF	Records details of employees
10	DEPARTMENT	Records details of the various departments
11	OFFICE	Records details of different office locations
12	MEMBERSHIP	Records details of customer membership, clubs, societies.
13	VEHICLE_SERVICE	Records details of different vehicle services offered
14	NOK	Records details of the next of kin
15	INSURANCE_COMPANY	Details of the Insurance organization giving the insurance cover
16	POLICY_RENEWABLE	Records details of due date of insurance policy
17	INCIDENT	Records details of the accident, theft, fire, etc.
18	INCIDENT_REPORT	Records details of the individual incident
19	COVERAGE	Records all terms and conditions in regard to the policy
20	PRODUCT	Records details of the products offered by insurance company
21	RECEIPT	Details of premium payments to customer
22	INSURANCE_POLICY_COVERAGE	It shows agreement and coverage details

Table 8: Abbreviation table of attributes manes used in LDM.

LDM Attributes	Column Name Abbreviations
CUST_ID	CUSTOMER_IDENTIFICATION
CUST_FNAME	CUSTOMER_FNAME
CUST_LNAME	CUSTOMER_LNAME
CUST_DOB	CUSTOMER_DATEOFBIRTH
CUST_PPS_NUMBER	CUSTOMER_PERSONALPUBLICNUMBER
STAFF_FNAME	STAFF_FNAME
STAFF_LNAME	STAFF_LNAME
STAFF_PPS_NUMBER	STAFF_PERSONALPUBLICNUMBER
ADMIN_COST	ADMINISTRATION_COST
NOK_ID	NEXTOFKIN_IDENTFICATION

Table-LDM 1: CUSTOMER

Attributes	Data type	Primary Keys and Foreign keys
CUST_ID	INT	PK
CUST_FNAME	VARCHAR (10)	
CUST_LNAME	VARCHAR (10)	
CUST_DOB	DATE	
CUST_GENDER	CHAR (2)	
CUST_ADDRESS	VARCHAR (20)	
CUST_MOB_NUMBER	VARCHAR (10)	
CUST_EMAIL	VARCHAR (20)	
CUST_PASSPORT_NUMBER	VARCHAR (20)	
CUST_MARITAL_STATUS	CHAR (10)	
CUST_PPS_NUMBER	INT	

EXPLANATION: -

The CUSTOMER attributes record all the essential personal details of the customer. The CUST_ID is the unique primary key.

Table-LDM 2: APPLICATION

Attributes	Data type	Primary Keys and foreign keys
APPLICATION_ID	VARCHAR (20)	PK
CUST_ID	INT	FK
VEHICLE_ID	INT	
APPLICATION_STATUS	CHAR (8)	
COVERAGE	VARCHAR (50)	

EXPLANATION: -

The APPLICATION attributes record all the essential application details of the customer. The APPLICATION_ID is the unique primary key and the CUST_ID is a foreign key linking the table back to the entity type CUSTOMER.

Table-LDM 3: QUOTE

Attributes	DATA TYPE	PRIMARY KEYS AND FOREIGN KEYS
QUOTE_ID	VARCHAR (20)	PK
APPLICATION_ID	VARCHAR (20)	FK
CUST_ID	INT	FK
ISSUE_DATE	DATE	
VALID_FROM_DATE	DATE	
VALID_TILL_DATE	DATE	
DESCRIPTION	VARCHAR (100)	
PRODUCT_ID	VARCHAR (20)	
COVERAGE_LEVEL	VARCHAR (20)	

EXPLANATION: -

The QUOTE attributes record all the essential quotation details of the customer. The QUOTE_ID is the unique primary key and APPLICATION_ID and CUST_ID is a foreign key linking the table back to the respective entities

Table-LDM 4: INSURANCE_POLICY

Attributes	**Data type**	**PRIMARY KEYS AND FOREIGN KEYS**
AGREEMENT_ID	VARCHAR(20)	PK
APPLICATION_ID	VARCHAR(20)	FK
CUST_ID	INT	FK
DEPARTMENT_NAME	VARCHAR(20)	
POLICY_NUMBER	VARCHAR(20)	
START_DATE	DATE	
EXPIRY_DATE	DATE	
TERMS_CONDITION_DESCRIPTION	VARCHAR(100)	

EXPLANATION: -

The INSURANCE POLICY attributes record all the essential policy details of the customer. The AGREEMENT_ID is the unique primary key and the CUST_ID, and APPLICATION_ID are linked to the other corresponding entities through their foreign keys.

Table-LDM 5: PREMIUM_PAYMENT

Attributes	Data type	PRIMARY KEYS AND FOREIGN KEYS
PREMIMUM_PAYMENT_ID	VARCHAR (20)	PK
CUST_ID	INT	FK
PREMIUM_PAYMENT_AMOUNT	INT	
PREMIUM_PAYMENT_SCHEDULE	DATE	
RECIEPT_ID	VARCHAR (20)	
POLICY_NUMBER	VARCHAR (20)	

EXPLANATION: -

The PREMIUM_PAYMENT attributes record all the essential policy premium payments details of the customer. The PREMIUM_PAYMENT_ID is the unique primary key and the CUST_ID is the foreign key linking tables to CUSTOMER entities.

Table-LDM 6: VEHICLE

ATTRIBUTES	Data type	PRIMARY KEYS AND FOREIGN KEYS
VEHICLE_ID CUST_ID POLICY_ID DEPENDENT_NOK_ID VEHICLE_REGISTRATION_NUMBER VEHICLE_VALUE VEHICLE_TYPE	INT	PK FK
	INT	
	VARCHAR (20)	
	VARCHAR (20)	
	VARCHAR (20)	
	INT	
	VARCHAR (20)	
VEHICLE_SIZE	INT	
VEHICLE_NUMBER_OF_SEAT	INT	
VEHICLE_MANUFACTURER	VARCHAR (20)	
VEHICLE_ENGINE_NUMBER	INT	
VEHICLE_CHASIS_NUMBER	INT	
VEHICLE_NUMBER	VARCHAR (20)	
VEHICLE_MODEL_NUMBER	VARCHAR (20)	

EXPLANATION: -

The VEHICLE attributes record all the essential VEHICLE details belonging to the customer. The VEHICLE_ID is the unique primary key and the CUST_ID is the foreign key linking table to CUSTOMER entity**.**

Table-LDM 7: CLAIM

ATTRIBUTES	Data type	PRIMARY KEYS AND FOREIGN KEYS
CLAIM_ID	INT	PK
CUST_ID	INT	FK
AGREEMENT_ID	VARCHAR (20)	
CLAIM_AMOUNT	INT	
INCIDENT_ID	VARCHAR (20)	
DAMAGE_TYPE	VARCHAR (20)	
DATE_OF CLAIM	DATE	
CLAIM_STATUS	CHAR (10)	

EXPLANATION: -

The CLAIM attributes record all the essential CLAIM details of the customer in case of an incident. The CLAIM_ID is the unique primary key and the CUST_ID is foreign key linking table to CUSTOMER entity.

Table-LDM 8:   CLAIM_SETTLEMENT

Attributes	Data type	PRIMARY KEYS AND FOREIGN KEYS
CLAIM_SETTLEMENT_ID	INT	PK
CUST_ID	INT	FK
VEHICLE_ID	INT	
DATE_SETTLED	DATE	
AMOUNT_PAID	INT	
COVERAGE_ID	VARCHAR (20)	
CLAIM_ID	INT	FK

EXPLANATION: -

The CLAIM_SETTLEMENT attributes record all the essential claim settlement details of the customer after an incident. The CLAIM_SETTLEMENT_ID is the unique primary key and the CUST_ID and CLAIM_ID are the foreign keys that link the table to the corresponding entity.

Table-LDM 9: STAFF

Attributes	Data type	PRIMARY KEYS AND FOREIGN KEYS
STAFF_ID	VARCHAR (20)	PK
COMPANY_NAME	VARCHAR (20)	FK
STAFF_FNAME	VARCHAR (10)	
STAFF_LNAME	VARCHAR (10)	
STAFF_ADDRESS	VARCHAR (20)	
STAFF_CONTACY	VARCHAR (10)	
STAFF_GENDER	CHAR (2)	
STAFF_MARITAL_STATUS	CHAR (10)	
STAFF_NATIONALITY	CHAR (15)	
STAFF_QUALIFIACATION	VARCHAR (20)	
STAFF_ALLOWANCE	INT	
STAFF_PPS_NUMBER	INT	

EXPLANATION: -

The STAFF attributes record all the essential staff details working in the insurance company. The STAFF_ID is the unique primary key and the COMPANY_NAME is a foreign key linking the table back to the entity type COMPANY.

Table-LDM 10: DEPARTMENT

Attributes	Data type	PRIMARY KEYS AND FOREIGN KEYS
DEPARTMENT_NAME	VARCHAR (20)	FK, PK
COMPANY_NAME	VARCHAR (20)	FK
OFFICE	VARCHAR (18)	
CONTACT_INFORMATION	VARCHAR (30)	
DEPARTMENT_STAFF	VARCHAR (18)	
DEPARTMENT_LEADER	VARCHAR (18)	

EXPLANATION: -

The DEPARTMENT attributes record all the essential company department details within the insurance company. The DEPARTMENT_NAME is the unique primary key and foreign key and the COMPANY_NAME is a foreign key linking the table back to the entity type COMPANY.

Table-LDM 11: OFFICE

Attributes	Data type	PRIMARY KEYS AND FOREIGN KEYS
OFFICE_NAME	VARCHAR (20)	PK
DEPARTMENT_NAME	VARCHAR (20)	FK
COMPANY_NAME	VARCHAR (20)	FK
OFFICE_LEADER	VARCHAR (20)	
CONTACT_INFORMATION	VARCHAR (20)	
ADDRESS ADMIN_COST STAFF	VARCHAR (20)	
	INT	
	VARCHAR (50)	

EXPLANATION: -

The OFFICE attributes record all the essential office details within the insurance company. The OFFICE_NAME is the unique primary key and the DEPARTMENT_NAME and COMPANY_NAME are foreign keys linking the table back to the respective entity types.

Table-LDM 12: MEMBERSHIP

Attributes	**Data type**	**PRIMARY KEYS AND FOREIGN KEYS**
MEMBERSHIP_ID	VARCHAR (20)	PK
CUST_ID	INT	FK
MEMBERSHIP_TYPE	CHAR (15)	
ORGANISTAION_CONTACT	VARCHAR (20)	

EXPLANATION: -

The MEMBERSHIP attributes record all the essential membership details available for insured customers. The MEMBERSHIP_ID is the unique primary key and the CUST_ID is a foreign key linking the table back to the entity type CUSTOMER.

Table-LDM 13: VEHICLE_SERVICE

Attributes	Data type	PRIMARY KEYS AND FOREIGN KEYS
VEHICLE_SERVICE	VARCHAR (30)	PK
VEHICLE_ID	INT	FK
CUST_ID	INT	FK
VEHICLE_SERVICE_ADDRESS	VARCHAR (20)	
VEHICLE_SERVICE_CONTACT	VARCHAR (20)	
VEHICLE_SERVICE_	VARCHAR (20)	
VEHICLE_SERVICE_INCHARGE	VARCHAR (20)	
VEHICLE_SERVICE_TYPE	VARCHAR (20)	
COMPANY_NAME	VARCHAR (20)	

EXPLANATION: -

The VEHICLE_SERVICE attributes record all the essential vehicle services offered to insured customer details. The VEHICLE_SERVICE is the unique primary key and the CUST_ID and VEHICLE_ID are linked to the other corresponding entities.

Table-LDM 14: NOK

Attributes	Data type	PRIMARY KEYS AND FOREIGN KEYS
NOK_ID	VARCHAR (20)	PK
AGREEMENT_ID	VARCHAR (20)	FK
APPLICTAION_ID CUST_ID NOK_NAME NOK_ADRESS NOK_PHONE_NUMBER NOK_GENDER NOK_MARITAL_STATUS	VARCHAR (20)	FK
	INT	FK
	VARCHAR (20)	
	VARCHAR (20)	
	VARCHAR (10)	
	CHAR (2)	
	CHAR (10)	

EXPLANATION: -

The NOK attributes record information on the next of kin details. NOK_ID is the unique primary key here AGREEMENT_ID, APPLICATION_ID, and CUST_ID are foreign keys linking back information to their respective entities.

Table-LDM 15: INSURANCE_COMPANY

Attributes	Data type	PRIMARY KEYS AND FOREIGN KEYS
COMPANY_NAME	VARCHAR(20)	PK
COMAPNY_DEPARTMENT_NAME	VARCHAR(20)	PK
COMPANY_ADRESS	VARCHAR(20)	
COMPANY_CONTACT_NUMBER	VARCHAR(10)	
COMPANY_FAX	INT	
COMPANY_EMAIL	VARCHAR(20)	
COMPANY_WEBSITE	VARCHAR(20)	
COMPANY_LOCTAION	VARCHAR(20)	
COMPANY_OFFICE_NAME	VARCHAR(20)	

EXPLANATION:-

The INSURANCE COMPANY attributes record all the essential company details of the customer. The COMPANY_NAME and COMPANY_DEPARTMENT_NAME are the primary keys.

Table-LDM 16: POLICY_RENEWABLE

Attributes	Data type	PRIMARY KEYS AND FOREIGN KEYS
POLICY_RENEWABLE_ID	VARCHAR(20)	PK
AGREEMENT_ID	VARCHAR(20)	FK
APPLICATION_ID CUST_ID	VARCHAR(20)	FK
DATE_OF_RENEWAL	INT	FK
TYPE_OF_RENEWAL	DATE	
	CHAR(15)	

EXPLANATION:-

The POLICY RENEWABLE attributes record all the essential policy renewal details of the insured customer. The POLICY_RENEWABLE_ID is the unique primary key and the AGREEMENT_ID, APPLICATION_ID and CUST_ID are foreign keys linking the table back to the respective entities.

Table-LDM 17: INCIDENT

Attributes	Data type	PRIMARY KEYS AND FOREIGN KEYS
INCIDENT_ID	VARCHAR(20)	PK
INCIDENT_TYPE	VARCHAR(30)	
INCIDENT_DATE	DATE	
DESCRIPTION	VARCHAR(100)	

EXPLANATION:-

The INCIDENT attributes record all the essential incident details such as Accident and theft on the insured customer vehicle. The INCIDENT_ID is the unique primary key.

Table-LDM 18: INCIDENT_REPORT

Attributes	**Data type**	**PRIMARY KEYS AND FOREIGN KEYS**
INCIDENT_REPORT_ID	VARCHAR(20)	PK
INCIDENT_ID	VARCHAR(20)	FK
CUST_ID	INT	FK
INCIDENT_TYEP	CHAR(10)	
INCIDENT_INSPECTOR	VARCHAR(20)	
INCIDENT_COST	INT	
INCIDENT_REPORT_DESCRIPTION	VARCHAR(100)	

EXPLANATION:-

The INCIDENT_REPORT_ID attributes record all the essential incident occurrences on the customer vehicle. The INCIDENT_REPORT_ID is the unique primary key and the CUST_ID, AND INCIDENT_ID are foreign keys linking the table back to their respective entity types.

Table-LDM 19: COVERAGE

Attributes	Data type	PRIMARY KEYS AND FOREIGN KEYS
COVERAGE_ID	VARCHAR(20)	PK
COMPANY_NAME	VARCHAR(20)	FK
COVERAGE_AMOUNT	INT	
COVERAGE_TYPE	CHAR(15)	
COVERAGE_LEVEL	CHAR(10)	
PRODUCT_ID	VARCHAR(20)	
COVERAGE_DESCRIPTION	VARCHAR(100)	
COVERAGE_TERMS	VARCHAR(50)	

EXPLANATION:-

The COVERAGE attributes record all the essential coverage details of the insurance policy to the customer. The COVERAGE_ID is the unique primary key and the COMPANY_NAME is a foreign key linking the table back to the entity type COMPANY.

Table-LDM 20: PRODUCT

Attributes	Data type	PRIMARY KEYS AND FOREIGN KEYS
PRODUCT_NUMB ER	VARCHAR(20)	PK
COMPANY_NAME	VARCHAR(20)	FK
PRODUCT_TYPE	CHAR(15)	
PRODUCT_PRICE	INT	

EXPLANATION:-

The PRODUCT attributes record all the essential company products details offered by the Insurance company. The PRODUCT_NUMBER is the unique primary key and COMPANY_NAME is the foreign key linking table to COMPANY entities.

Table-LDM 21: RECEIPT

Attributes	Data type	PRIMARY KEYS AND FOREIGN KEYS
RECIEPT_ID	VARCHAR(20)	PK
PREMIUM_PAYMENT_ID	VARCHAR(20)	FK
CUST_ID	INT	FK
TIME	DATE	
COST	INT	

EXPLANATION:-

The RECEIPT attributes record all the essential payments done by CUSTOMERS to the Insurance company. The RECEIPT_ID is the unique primary key and PREMIUM_PAYMENT_ID and CUST_ID are Foreign keys linking tables to their respective entities.

Table-LDM 22: INSURANCE_POLICY_COVERAGE

Attributes	**Data type**	**PRIMARY KEYS AND FOREIGN KEYS**
AGREEMENT_ID	VARCHAR(20)	PK
COVERAGE_ID	VARCHAR(20)	FK

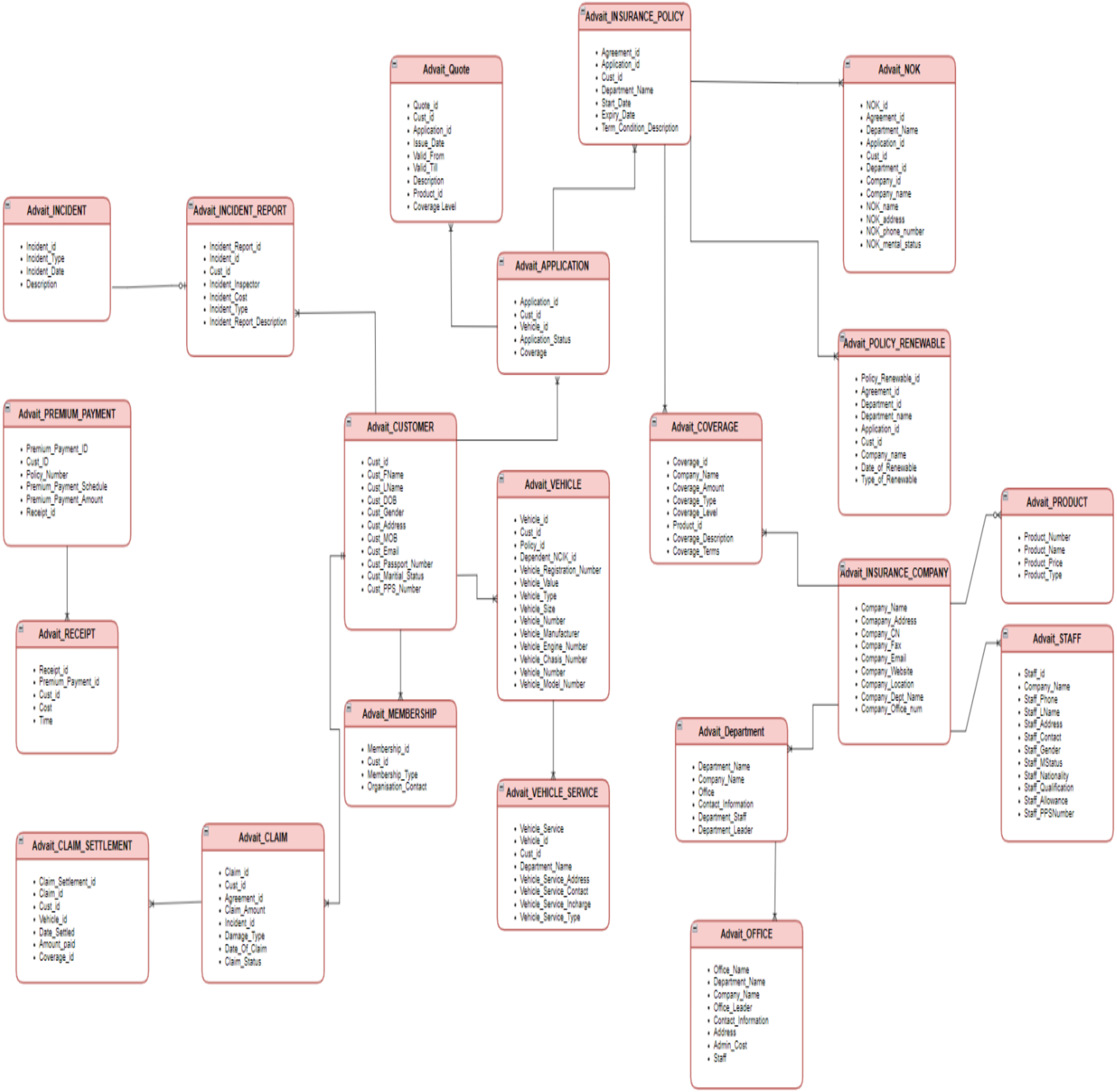
EXPLANATION:-

The INSURANCE_POLICY_COVERAGE records details of the Vehicle policy that entails Terms Conditions of the Contract.
AGREEMENT_ID is the unique primary key and COVERAGE_ID is the Foreign key linking this table to COVERAGE Entity.

Graphical presentation of LDM

The Logical Data Model (LDM) that we have designed for this part of report in graphical Figure-LDM 1. It has all the entity types, attributes and relationships that are valid and pertinent in designing our online vehicle insurance database system.

Figure-LDM 1: Logical data model of Vehicle insurance for AVIVA Ltd.



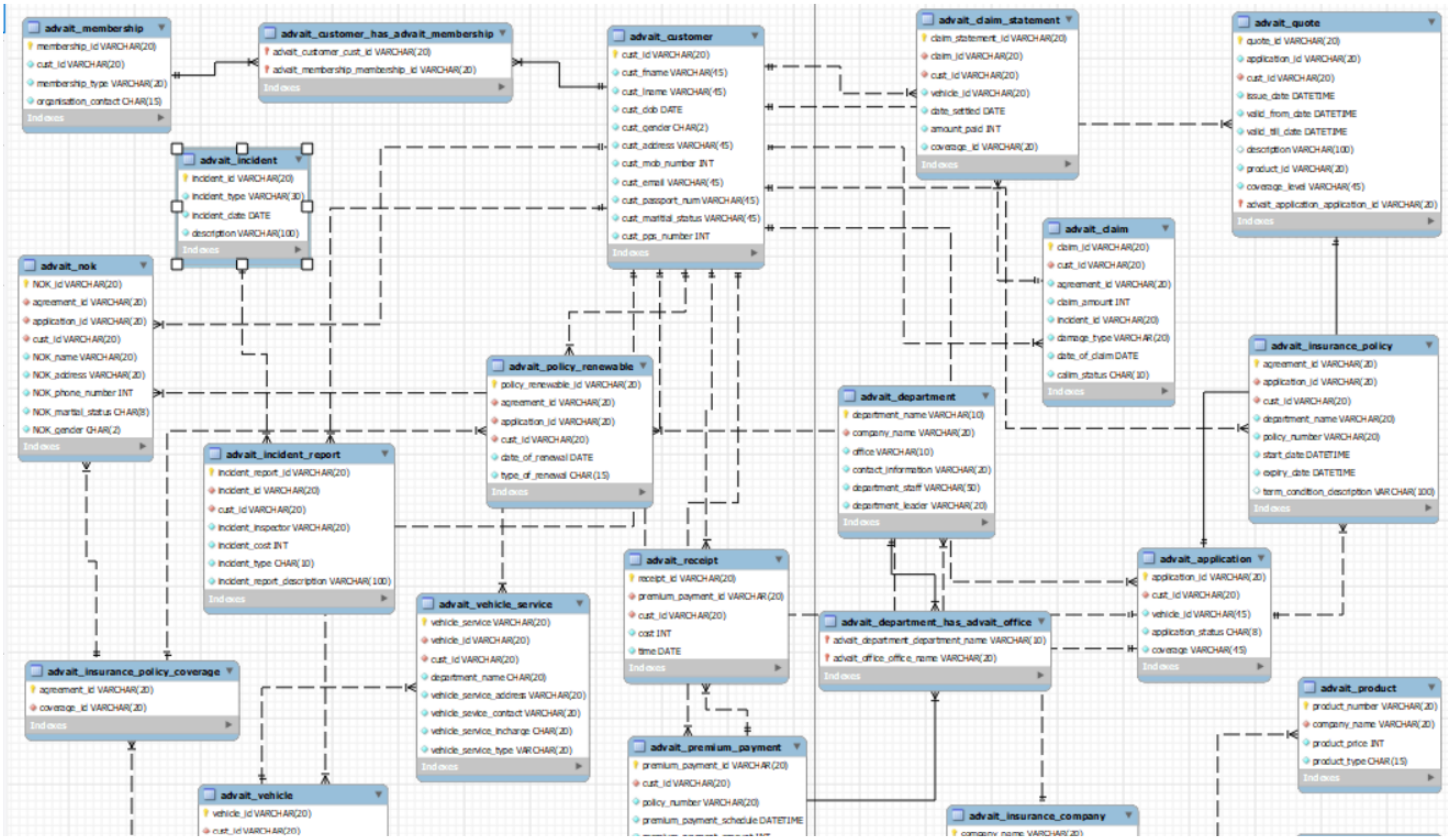
Part C

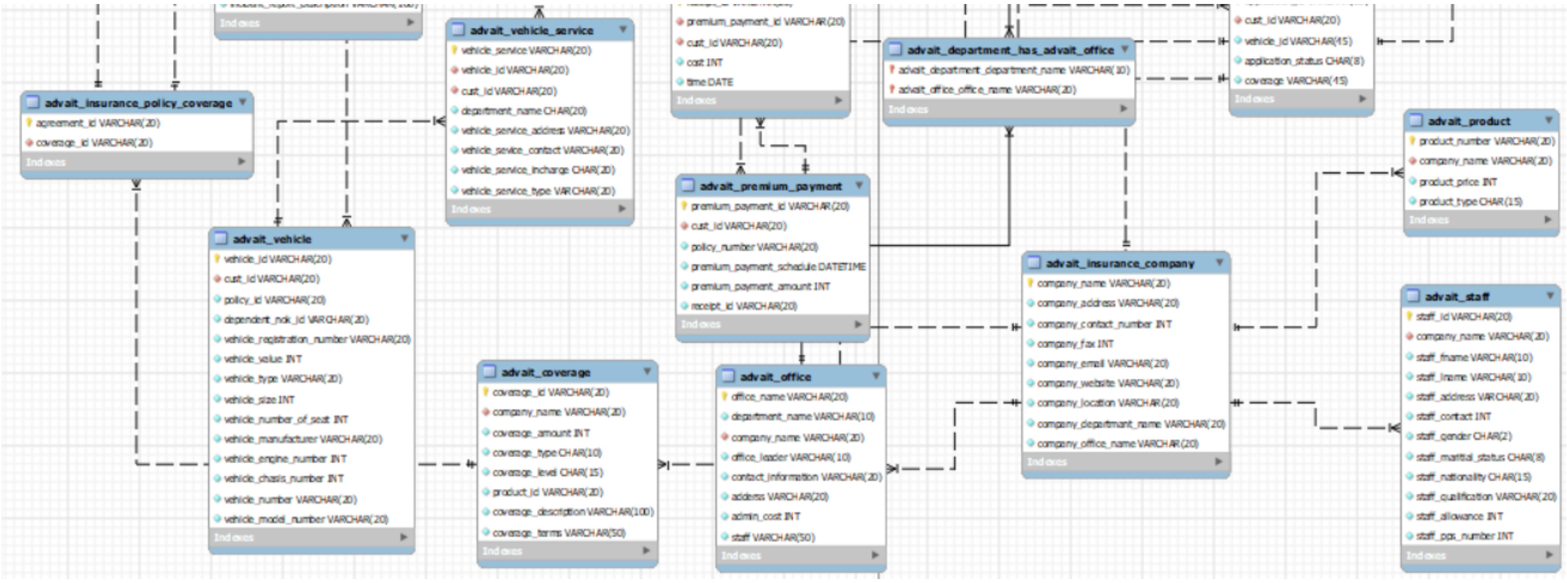
Physical Data Model

& Database Design

Introduction

When building an entity relationship (ER) model we tend to use it to later build different physical models of database types. Therefore, physical data model is used to implement into different technical software and hardware environments that is due to current state of technology and is changing as technologies change.




Indexes
advait_insurance_policy_coverage
agreement_id VARCHAR(20)
coverage_id VARCHAR(20)
Indexes
advait_vehicle_service
vehicle_service VARCHAR(20)
vehicle_id VARCHAR(20)
cust_id VARCHAR(20)
department_name CHAR(20)
vehicle_service_address VARCHAR(20)
vehicle_sevice_contact VARCHAR(20)
vehicle_service_incharge CHAR(20)
vehicle_service_type VARCHAR(20)
Indexes
premium_payment_id VARCHAR(20)
cust_id VARCHAR(20)
cost INT
time DATE
Indexes
advait_department_has_advait_office
advait_department_department_name VARCHAR(10)
advait_office_office_name VARCHAR(20)
Indexes
cust_id VARCHAR(20)
vehicle_id VARCHAR(45)
application_status CHAR(8)
coverage VARCHAR(45)
Indexes
advait_product
product_number VARCHAR(20)
company_name VARCHAR(20)
product_price INT
product_type CHAR(15)
Indexes
advait_vehicle
vehicle_id VARCHAR(20)
cust_id VARCHAR(20)
policy_id VARCHAR(20)
dependent_nok_id VARCHAR(20)
vehicle_registration_number VARCHAR(20)
vehicle_value INT
vehicle_type VARCHAR(20)
vehicle_size INT
vehicle_number_of_seat INT
vehicle_manufacturer VARCHAR(20)
vehicle_engine_number INT
vehicle_chasis_number INT
vehicle_number VARCHAR(20)
vehicle_model_number VARCHAR(20)
Indexes
advait_premium_payment
premium_payment_id VARCHAR(20)
cust_id VARCHAR(20)
policy_number VARCHAR(20)
premium_payment_schedule DATETIME
premium_payment_amount INT
receipt_id VARCHAR(20)
Indexes
advait_insurance_company
company_name VARCHAR(20)
company_address VARCHAR(20)
company_contact_number INT
company_fax INT
company_email VARCHAR(20)
company_website VARCHAR(20)
company_location VARCHAR(20)
company_departmant_name VARCHAR(20)
company_office_name VARCHAR(20)
Indexes
advait_staff
staff_id VARCHAR(20)
company_name VARCHAR(20)
staff_fname VARCHAR(10)
staff_lname VARCHAR(10)
staff_address VARCHAR(20)
staff_contact INT
staff_gender CHAR(2)
staff_maritial_status CHAR(8)
staff_nationality CHAR(15)
staff_qualification VARCHAR(20)
staff_allowance INT
staff_pps_number INT
Indexes
advait_coverage
coverage_id VARCHAR(20)
company_name VARCHAR(20)
coverage_amount INT
coverage_type CHAR(10)
coverage_level CHAR(15)
product_id VARCHAR(20)
coverage_description VARCHAR(100)
coverage_terms VARCHAR(50)
Indexes
advait_office
office_name VARCHAR(20)
department_name VARCHAR(10)
company_name VARCHAR(20)
office_leader VARCHAR(10)
contact_information VARCHAR(20)
address VARCHAR(20)
admin_cost INT
staff VARCHAR(50)
Indexes

Normalization

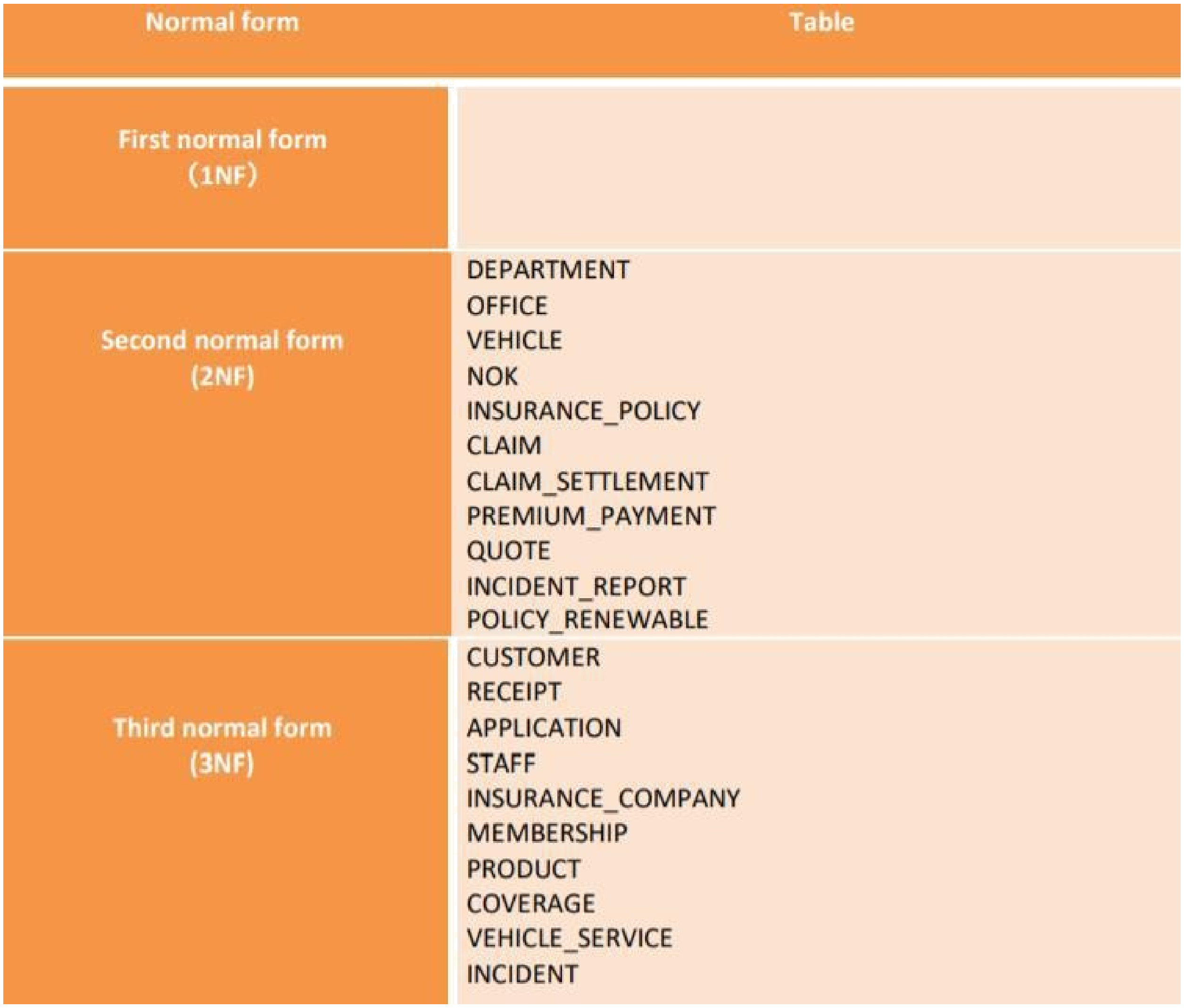
When building an entity relationship (ER) model we tend to use it to later build different physical models of database types. Therefore, physical data model is used to implement into different technical software and hardware environments that is due to current state of technology and is changing as technologies change.

1. First Normal Form (1NF):
If a relation contains composite or multi-valued attribute, it violates first normal form, or a relation is in first normal form if it does not contain any **composite** or **multi-valued attribute**. A relation is in first normal form if every attribute in that relation is singled valued attribute.

2. Second Normal Form (2NF):
Second Normal Form (2NF) is based on the concept of full functional dependency. To be in second normal form, a relation must be in first normal form and relation must not contain any partial dependency.

3. Third Normal Form (3NF):
A relation is in third normal form, if there is no transitive dependency for non-prime attributes as well as it is in second normal form.

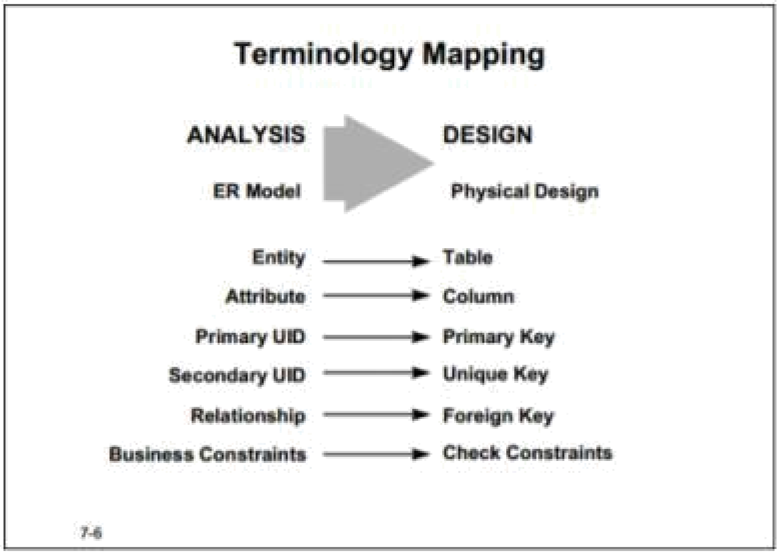
Normalization of Vehicle Insurance Company



Normal form	Table
First normal form (1NF)	
Second normal form (2NF)	DEPARTMENT OFFICE VEHICLE NOK INSURANCE_POLICY CLAIM CLAIM_SETTLEMENT PREMIUM_PAYMENT QUOTE INCIDENT_REPORT POLICY_RENEWABLE
Third normal form (3NF)	CUSTOMER RECEIPT APPLICATION STAFF INSURANCE_COMPANY MEMBERSHIP PRODUCT COVERAGE VEHICLE_SERVICE INCIDENT

Terminology

The relational data model consists of mathematical rules, that later translate its syntax to physical model, but there is not always correspondent to syntax of physical model. Therefore, you have to keep track of them and find how to implement them in correct way. This change also means change of terminology.



Terminology Mapping

ANALYSIS	DESIGN
ER Model	Physical Design
Entity	Table
Attribute	Column
Primary UID	Primary Key
Secondary UID	Unique Key
Relationship	Foreign Key
Business Constraints	Check Constraints

7-6

RDBMS

As we start doing to build relational data management system (RDBMS) we need a large number of parameters to obtain a correct adapted physical model. You must be aware that there is no absolute truth here. Some of most important points of creating physical models for RDBMS :

- Expected volume of tables, the hardware characteristics (CPU speed, memory size, number of disks and corresponding space), the architecture- client/server or three size, the network bandwidth, speed and operating systems are important determinants.
- User experience second big issue (response time, the GUI and frequency of use of modules).
- Depending which version, you use as some elements may or may not exist.

PHYSICAL MODEL

When creating physical models, we create tables or clusters and we must write specifications of internal data type for each its columns. These types define generic domain of values that each column can contain.

Some concerns using data types building a physical model:

- Data types can have a narrow focus (number, date).
- Some are general purpose data types (various character data types).
- Data types can allow for variable length or not. Choosing a large fixed length for a column to only store a few bytes per row, makes a large table. This may affect performance specially if stored on multiple blocks, resulting in great number of I/O's and so affecting performance.
- Large data object types are not advised to be used in where clause as they are only retrievable against other columns

The Data types used in our project:

- **CHAR (size)** these are fixed-length character data of length-sized bytes. Maximum size is 2000 bytes.

  Typical use: for official International Currency Codes which are a fixed three characters in length such as USD, FFR.

- **VARCHAR (size)** Variable-length character string having maximum length-sized bytes. Maximum size is 4000, and minimum is 1. This is the most commonly-used data type and you should use it if you are not sure which one to use.

  Typical use: for storing individual ASCII text lines of unlimited length ASCII texts on which you need to be able to search using a wildcard.

- **DATE** Valid date range from January 1, 4712 BC to December 31, 4712 AD. A date data type also contains time components. You should use it only when you know the full date including day, month, and year. The time component is often set to 00:00 (midnight) in normal use of dates.

Typical use: any date where the full date is known.

- **INTEGER:** The INTEGER datatype is usually referred to as NUMBER(38). Its precision can range from 1 to 38.

DATABASE SECURITY

Database security is described by the following aspects：

- Data independence

Data independence is an important method ensures data security; it can be divided into logical independence and physical independence. Physical independence means applications and data are store independent of each other data is managed by DBMS and application not need understand it, application just need processing the data when the physical storage of data changing application without changing.

- Data Security

1. Isolation protect important files in database.
2. Using authorization rules, such as access control method and accounts, passwords permissions control.
3. Data Encryption

- Data Integrity

1. Data validation: ensure clean, correct and useful data.

2. Data consistency: Different users are using the same data should be identical.

3. Data correctness: The input value of the data should be consistent with

data in database.

- Concurrency Control

Concurrency Control means the database is a shared resource for multi-use, When multiple user concurrent access to data, in the database will have multiple transactions simultaneously access the same data. If not controlled for concurrent operation may be cause incorrect to read and store data, destroy the consistency of the database.
Locking mechanism of the database can effectively protect the database achieve concurrency control.

The database provides three different types of locks:

1. DML lock: DML locks used to protect data integrity; DML locks mainly include TM lock and TX lock. TM lock called table lock, TX called transaction locks or row locks.

2. DDL lock: DDL lock protects the structure of the database objects. In the DDL operation is automatically added DDL lock to the object, to protect these objects will not be modified by other sessions.

3. Internal locks and Latches: Protect the internal structure of the database.

- Recovery

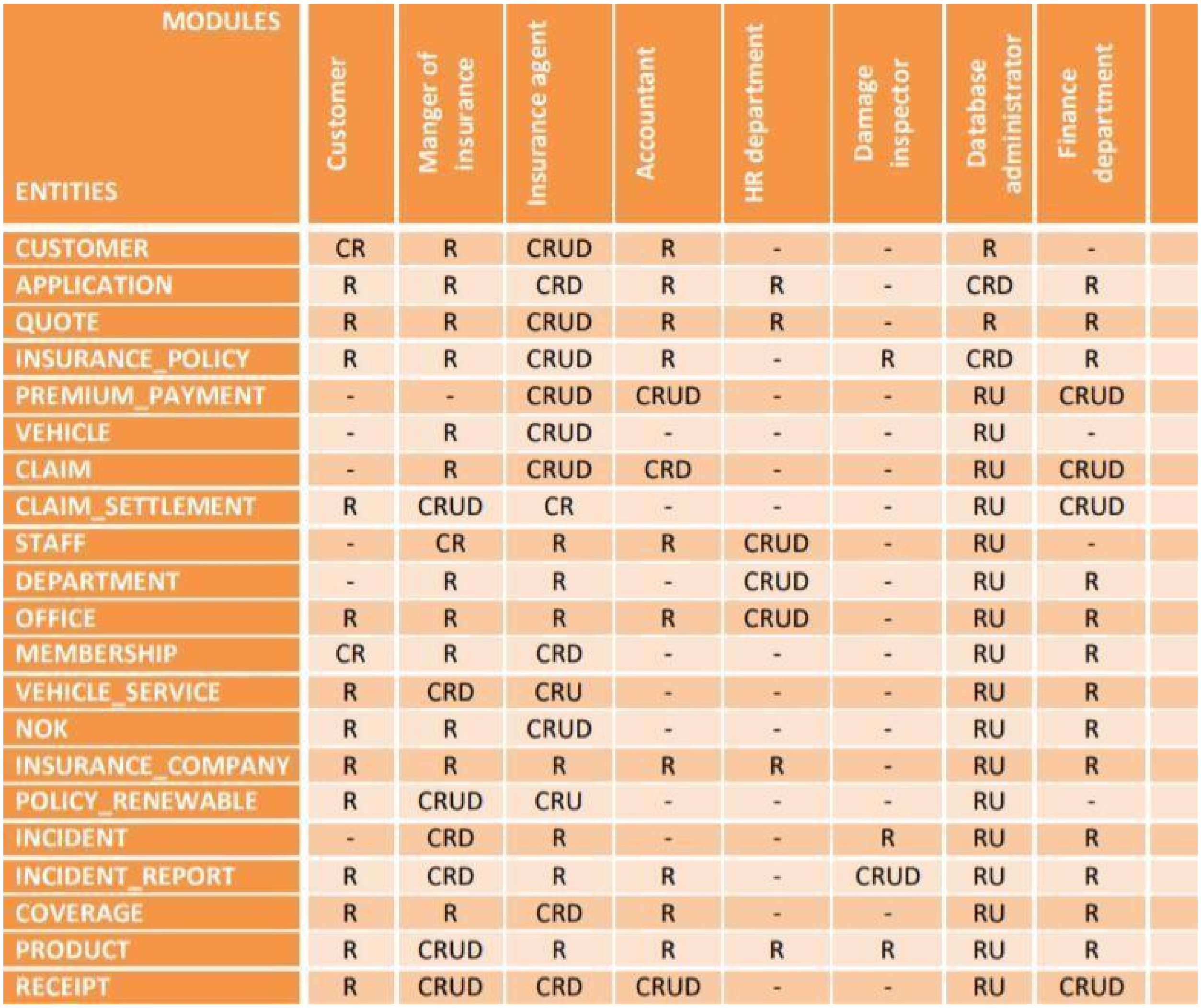
When the database fails DBMS need to find faults and fix problems, thus preventing data corruption. Moreover, database should regularly back up and establish a spare machine, makes the database can be restored as quickly as possible from the fault.

Who has access to certain parts of database ?

“A CRUD matrix is a table showing the functions in an application containing SQL statement affecting parts of a database” (Williams, 2001). It is a great way to show us what kind interaction appears between user and tables in database. This analysis of possible user scenarios and shows us tables that are very used and those that are not used, and it brings us the view how database is burdened with possible – bottleneck in system performance (Williams, 2001).

Using four SQL statements:

- Create – INSERTE – to store new data
- Read – SELECT – to retrieve data
- Update – UPDATE – to change or modify data
- Delete – DELETE – delete or remove data



MODULES ENTITIES	Customer	Manger of insurance	Insurance agent	Accountant	HR department	Damage inspector	Database administrator	Finance department	
CUSTOMER	CR	R	CRUD	R	-	-	R	-	
APPLICATION	R	R	CRD	R	R	-	CRD	R	
QUOTE	R	R	CRUD	R	R	-	R	R	
INSURANCE_POLICY	R	R	CRUD	R	-	R	CRD	R	
PREMIUM_PAYMENT	-	-	CRUD	CRUD	-	-	RU	CRUD	
VEHICLE	-	R	CRUD	-	-	-	RU	-	
CLAIM	-	R	CRUD	CRD	-	-	RU	CRUD	
CLAIM_SETTLEMENT	R	CRUD	CR	-	-	-	RU	CRUD	
STAFF	-	CR	R	R	CRUD	-	RU	-	
DEPARTMENT	-	R	R	-	CRUD	-	RU	R	
OFFICE	R	R	R	R	CRUD	-	RU	R	
MEMBERSHIP	CR	R	CRD	-	-	-	RU	R	
VEHICLE_SERVICE	R	CRD	CRU	-	-	-	RU	R	
NOK	R	R	CRUD	-	-	-	RU	R	
INSURANCE_COMPANY	R	R	R	R	R	-	RU	R	
POLICY_RENEWABLE	R	CRUD	CRU	-	-	-	RU	-	
INCIDENT	-	CRD	R	-	-	R	RU	R	
INCIDENT_REPORT	R	CRD	R	R	-	CRUD	RU	R	
COVERAGE	R	R	CRD	R	-	-	RU	R	
PRODUCT	R	CRUD	R	R	R	R	RU	R	
RECEIPT	R	CRUD	CRD	CRUD	-	-	RU	CRUD	

ADD QUERY WITH EXPLAINATION

1. Retrieve Customer and Vehicle details who has been involved in an incident and claim status is pending – Customer, vehicle, claim status, incident.

```
SELECT c.cust_id, v.vehicle_id FROM ADVAIT_customer C, ADVAIT_vehicle V WHERE
V.CUST_ID = C.CUST_ID AND
        C.CUST_ID IN ( SELECT C1.CUST_ID FROM ADVAIT_claim C1 WHERE C1.claim_status
  = 'PENDING' AND C1.INCIDENT_ID IN (SELECT I.INCIDENT_ID FROM
ADVAIT_incident_report I
  WHERE I.CUST_ID = C1.CUST_ID AND I.vehicle_id = V.vehicle_id ));
```

2. Retrieve customer details who has premium payment amount greater than the sum of all the customerIds in the database – premium payment, customer

```
select c.cust_id from advait_customer as c
where c.cust_id in (
        select cust_id from advait_premium_payment where
        premium_payment_amount > (
                select sum(cust_id) from advait_customer));
```

3. Retrieve Company details whose number of products is greater than departments, where the departments are located in more than one location—company, product, departments, office

```
SELECT C.COMPANY_NAME FROM advait_INSURANCE_COMPANY C INNER JOIN
advait_PRODUCT P ON P.COMPANY_NAME=C.COMPANY_NAME GROUP BY
P.COMPANY_NAME
HAVING COUNT(*) >ALL(SELECT COUNT(*) FROM advait_INSURANCE_COMPANY GROUP
BY COMPANY_NAME HAVING COUNT(COMPANY_LOCATION)>1);
```

4. Select Customers who have more than one Vehicle, where the premium for one of the Vehicles is not paid and it is involved in accident

```
SELECT CONCAT(C.CUST_ID,",",CUST_FNAME,CUST_LNAME) AS CUST_INFO FROM
(((advait_CUSTOMER C INNER JOIN advait_VEHICLE V ON (C.CUST_ID=V.CUST_ID) )
  INNER JOIN advait_PREMIUM_PAYMENT P ON (P.CUST_ID=V.CUST_ID AND
P.PREMIUM_PAYMENT_AMOUNT=0)) INNER JOIN advait_INCIDENT_REPORT IR ON
(IR.CUST_ID=P.CUST_ID AND IR.INCIDENT_TYPE='ACCIDENT'))
  GROUP BY V.CUST_ID HAVING COUNT(V.VEHICLE_ID)>1;
```

5. Select all vehicles which have premium more than its vehicle number.

```
SELECT distinct(V.VEHICLE_ID) FROM advait_VEHICLE V, advait_PREMIUM_PAYMENT P
```

```
WHERE
        V.CUST_ID = P.CUST_ID AND P.PREMIUM_PAYMENT_ID > V.VEHICLE_NUMBER;
```

6. Retrieve Customer details whose Claim Amount is less than Coverage Amount and Claim Amount is greater than Sum of (CLAIM_SETTLEMENT_ID, VEHICLE_ID, CLAIM_ID, CUST_ID)

```
SELECT CONCAT(C.CUST_ID,",",CUST_FNAME,",",CUST_LNAME) AS CUST_INFO FROM
(advait_CUSTOMER C INNER JOIN advait_CLAIM_SETTLEMENT CS ON
CS.CUST_ID=C.CUST_ID
INNER JOIN advait_CLAIM CL ON (CL.CLAIM_ID=CS.CLAIM_ID)INNER JOIN
advait_COVERAGE CO ON (CO.COVERAGE_ID=CS.COVERAGE_ID AND
CL.CLAIM_AMOUNT<CO.COVERAGE_AMOUNT AND
CL.CLAIM_AMOUNT>(CS.CLAIM_SETTLEMENT_ID+CS.VEHICLE_ID+CS.CLAIM_ID+CS.CUST_
ID)));
```

Conclusion:

We have a class project created database (DB) with all documentations and reports included. Our goal was to create DB for Online vehicle insurance company with code. There were some big and small challenges but we succeeded in making a functional DB. We started to build conceptual data model (CDM) we continued with logical data model (LDM) and then we made physical data model (PDM) all in Erwin software program.