Introduction to Databases

Final Project

Website URL : disabled

E. May
6-12-2018

Database Outline

This database project is based on the insurance industry from agency management perspective. An agency management system is basically a customer account management system where all transactions and contract details are stored. The easiest way to think about this topic is as a representation of the purchasing path/process of insurance. It starts with the insurance company, that works with a local agent, who sells to a customer a policy which has various coverages.

Another way to think of it is as the insurance company is the manufacturer of a product, the policy, which is made up of parts, coverages. This product (the policy) is then sold by an agent (think retailer) to a customer locally.

Since the insurance industry can get a bit complex, I have simplified as much as possible and adjusted some details to fit this database project. My previous career as a commercial underwriter for a large insurance company gives me an insider's perspective but can also make it hard for me to see the areas that outsiders may get confused by. With that in mind, I have made as many notes as possible on my thought process and am available for questions should any confusion arise.

Entities:

Customers (Insureds) – The customer purchasing the insurance policy. My database will focus on commercial accounts and therefore the customers will be businesses like Apple Inc or Johnson & Smith LLC. The customer table records the majority of the general information about a customer, including:

- **CID** – Short of Customer ID, this number is automatically assigned to each customer when they are recorded in the database. It is an auto-incrementing integer and serves as the primary key for customers.
- **Co_Name** – The full company name of the customer (since the customers are commercial businesses). This is a variable character string of maximum 50 characters.
- **website** – Company website URL (100 max characters), if available but defaults to null.
- **address_id** – A foreign key which references an address separately recorded in the Address table. FK integer that cannot be null.
- **phone** – The primary customer phone number, but since everyone types in phone numbers differently, it is a variable character string of up to 20 characters in length to account for dashes and international numbers. It cannot be null as an agent must have some way to contact the company.
- **p_contact** – Primary contact for the customer (which again is a commercial business). This is now a simple VARCHAR to simplify database. Normally I would store this in a a more detailed structure, but having it be a simple variable text field lets the user decide who and what is most important to keep track of. If they only know that John is the contact or it's the CEO they can record whichever info they have that fits in 35 characters.
- **op_type** – Foreign key link to an integer id refencing an operations type, separately recorded in the 'operations' table.

- **op_desc** – This is a freeform TEXT field which can be used to record notes about the unique nature of the customers specific operations. It is not required, but usually the agent has/needs more detail about an operation than simply it being a manufacture, for instance what type (toy or computer or widget) or an explanation of a more complicated business.

__Carriers__ – A carrier is what the insurance industry calls insurance companies which provide (issues/makes) the insurance policy. For this database the only information we need about the carrier is the name, other attributes/details would just be filler that often isn't even captured in an agency management system because it is stored elsewhere.

- **Carrier ID** – Primary identification key for the insurance carriers. This is an auto-generated, unsigned integer and cannot be null.
- **Carrier Name** – The full official name of the insurance company ("Liberty Mutual"). It is a variable character string (up to 50 characters in length) and cannot be null.

__Agents Staff__ – The agents are the middle-man distributors of the policy from the carrier to the customer. Since the real data work is done by account managers and there are other assistants in the process, I am tracking the parties involved as Staff. The relationship of a staff member assigned to an account will be represented separately in its own table.

- **Staff_id** – This the number automatically assigned (& auto incremented) to each staff member when recorded in the database. It servers as the primary key to identify the staff within the table.
- **fname** – First name of the staff member which is a string of maximum 50 characters. IT cannot be blank and there is no default.
- **lname** – Last name of the staff member recorded in a string of maximum 50 characters. It cannot be blank and there is no default.
- **role_id** – Foreign key referencing a role id/title separately recorded. (ID for agent or acct. mgr. etc).
- **email** – String to record the staff email addresses which defaults to null if not provided. Maximum 50 characters.
- **active** – T/F Boolean status indicating if the staff member is currently active or is a past employee. Cannot be null and defaults to True.

__Policies__ – The insurance policy is a contract that provides coverage for a specific term and premium, from a carrier to an insured. (Example: Home or Auto Insurance from Liberty Mutual for 1 year, premium - $1,000) **Probably the most important entity and data, because it is a relationship in itself. It is __the__ product everything else is supporting.**

- **PID** – Policy ID number which is automatically assigned to the policy when it is recorded in the agency database. The auto-incrementing integer is a primary key
- **Policy_no** – The policy number is the carrier assigned identification number of a contract/policy. (Think of it as a serial number provided by the manufacturer) It is an extremely important if not the most important piece of information which cannot be null. It will be a mix of numbers & letters

(EB2-34H34K) and each carrier uses a different format & length, so the field is a string of up to 20 characters max.

- **Acct_id** – The foreign key which references the customer account which is insured (owns/bought) the policy. Basically tells whose policy it is. As such it is required and cannot be null.
- **Lob_id** – Foreign key which references what line of business the policy is (type). Cannot be null
- **Premium** – The policy premium dollar amount. Can range from a few hundred dollars to millions. Cannot be null and is recorded to the decimal.
- **Carrier_id** – Foreign key referencing the carrier (insurance company) that writes/issues the policy.
- **Status** – Most account managers keep track of what stage there work (policies) is in. But since everyone labels stuff differently (renewing, processing, editing etc.) it is a short variable string that can be entered or not, as needed. Maximum 25 characters.
- **i_date** – The inception date of the policy term (when it starts). Extremely important to know when a policy gives coverage and when it no longer does.
- **x_date** – The expiration date of the policy term (when it ends). Extremely important and is typically the same month and day as the inception, just one year later.

<u>Coverage</u> –There are many different types of insurance coverage. These coverages can be thought of as the terms of the insurance contract. This database will focus on commercial policies like General Liability, Property, Work Comp, Errors & Omissions etc . Coverages are specific parts of a policy and need to be separated, but recording this info is not mandatory. Recording all the details of a policy can be time-consuming so agents/account managers typically only record the highlights and if more information is needed, they can consult the policy contract directly (pdf).

- **Covg_id** – The automatically assigned number when a type of coverage is created. Must be unique auto-incrementing, and not null primary key.
- **Covg_type** – A string description of the coverage term (“bodily injury” or “building”) which is a maximum of 100 characters and cannot be null.

<u>Contacts</u> – The people/employee that represent the customer (remember, in this case a company) are considered contacts. Since a company is just a legal entity it needs people to communicate for it and it is important to record the parties specifically involved. These contacts are typically the CEO or CFO or Owner, but regardless a company needs to have a primary representative contact. Unfortunately, it isn’t always easy to keep track of all the contacts for a particular account, so only the primary account contact is required for a customer record, but any contact can be entered without attaching to a specific customer. ***Note the intention is to make as few of the attributes mandatory as possible so that a contact gets recorded even if not all of their information is remembered.***

- **Contact_id** – The primary id assigned to a contact when recorded. Should be an auto-incrementing integer which cannot be null like other primary keys.
- **Acct_id –** A foreign key reference to the ID of a customer (account).
- **Fname** – First name of contact which is a string of maximum 50 characters. It cannot be blank and there is no default.
- **Lname** – Last name of contact is a string of maximum 50 characters, which while it seems weird can be blank because we want to record even informal contacts if necessary.

- **Title** – The title (position, role, occupation) of the contact person so that we know if we are talking about a CEO or an accountant. Recorded as a character string (max 50 characters) and defaults to null.
- **Email** – Another important detail to have but not is not mandatory, is email. This detail is
- **Phone** – Another important detail to have but not is not mandatory, is phone. This detail is

Other secondary entities are used to organize and standardize repetitive date. They include:

LOB – Line of Business (LOB) is the type of insurance policy. Examples would be Package, Auto, Work Comp, Home, Umbrella, Health.

- **Lob_id** – Auto-incrementing number which is assigned when a we record an line of business type in the database. Cannot be null and is the primary key.
- **Lob_type** – The primary type of a policy

Address – Since an address incorporates multiple smaller pieces of data, it is good to structure that separately in its own table so it can be searched and formatted as needed. I also want to note that addresses are extremely important in insurance because they are reference in policies and knowing where something is, goes into the risk and pricing evaluation. Insurance companies often want to know what county a building is in to look up local codes, fire department info and all sorts of things. This is why the address is broken down to the smallest level of attributes.

- **Ad_id** – Auto-incrementing number which is assigned when a we record an address in the database. Cannot be null and is the primary key.
- **Address1 & Address2** - First & second line of an street address recorded as a string no longer than 50 characters and only first line (Address1) is required (not null).
- **City** – City should fit in a variable length string (50 character max) which cannot be null.
- **County** – County is a helpful piece of information for insurance and is recorded if available as a character string (50 characters max) and defaults to null.
- **Sstate** – 3 character short abbreviation for a state (OR, WA, CA) which can be null if outside USA. (will be creating that criteria later if possible).
- **Zip** – Since not all zip codes are number (international) it is a variable character string of up to 15 characters and defaults to null.
- **Country** – The country is recorded in a 30 character field which cannot be null. This allows for either country abbreviation like USA or long formal names like Canada.

Roles – As I mentioned earlier, the "agent" is not the only person servicing an customer account. Account managers,claims adjustors, underwriters, agents/producers, assistants are all assigned to accounts as needed and knowing what role they play is important to managing the work load.

- **Role_id** – The primary key, unique identifier (integer) of a type of staff role. Cannot be null and is auto-incrementing small integer (since there are not that many role types).
- **Role_title** – The role title (manager, agent etc.) is a string that of not more than 25 characters that cannot be null.

<u>Operations</u> – Operations are the standard descriptions of what a business does. In this case they are based on the nationally standardized codes (SIC) that the Department of Labor has created. For example operations code 42 is for Wholesale Trade & 23 is Construction. While the codes get more detailed, for this database I am keeping it on a high level and not distinguishing between a apple farm & a wheat farm, but labeling both as agricultural so I can demonstrate searching techniques (sql queries) without having thousands and thousands of records that.

- **Ops_code** – The operations code will be the SIC code already pre-assigned by the government to be unique. This primary key integer cannot be null and is not auto-incrementing because new ops_codes will not be created to unless the government creates a new one with an already determined ID/code.
- **Ops_desc** – The standard description of an operation. It is a string of variable length (maximum 50 characters).

Many to Many Relationships explored later in this report s are represented as tables in this database but on their own are not entities. Their attributes are simply the foreign key references and primary key to identify the unique relationship.

<u>Other Entity & Attribute Notes:</u>

- A few of the entities have a last-updated attribute to keep track of when records are edited and such. The attribute is the same in each case it is used defined as: TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP.
- I tried to show a variety of methods to record data, but I may change some of the types as we progress in this project.
- For the ER Diagram I used the common names of attributes (not specific lables like CID or ops_desc) to make it more read-able.

Relationships:

<u>**Customers & Policies**</u> – Customers typically have many policies (at least one), but each policy can only have one insured (customer). No two customers can have the exact same policy, each policy is unique. (Many to One)

<u>**Agency Staff & Customers**</u> – Agents serve/sell to many customers and in some cases customers may have multiple agents or account managers work with them depending on the policy situation. There is always at least one staff member (usually agent) assigned to a Customer. (Many to Many)

<u>**Policies & Carriers**</u>– Each policy can be issued by only one carrier, but carriers have many policies issued to different or sometimes the same insureds. (One to many)

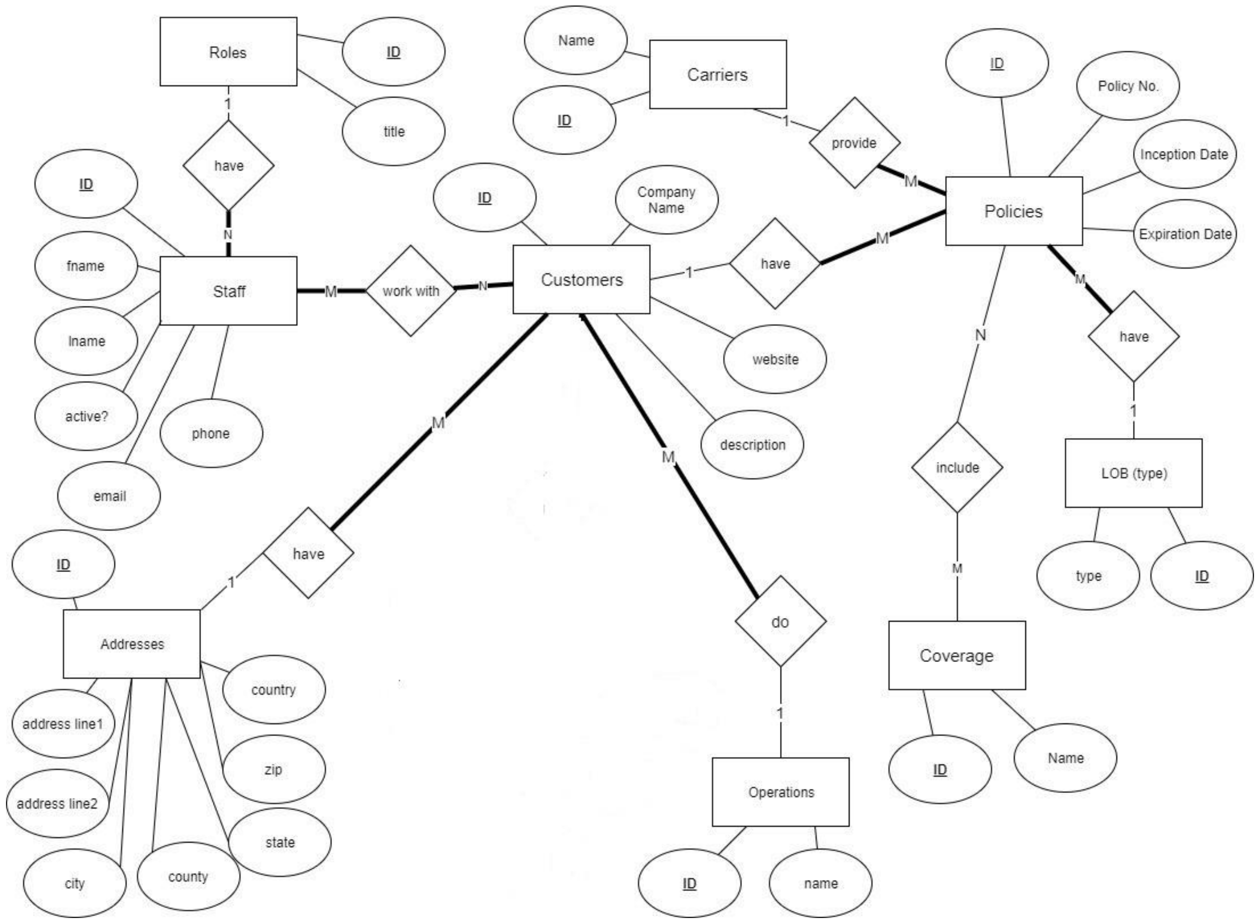
<u>**Policy & Coverage**</u> – Each policy is made up of coverages and each type of coverage can be used on many different policies. There is no mandatory participation. (Many to Many)

<u>**Carrier & Customers**</u> – Carriers have many insureds (customers of policies) and insureds often purchase insurance from more than carrier, so many carriers can have the same insured. But since the carrier & customer relationship is its own entity, a policy, this is not directly represented in the database diagrams. (Many to Many)

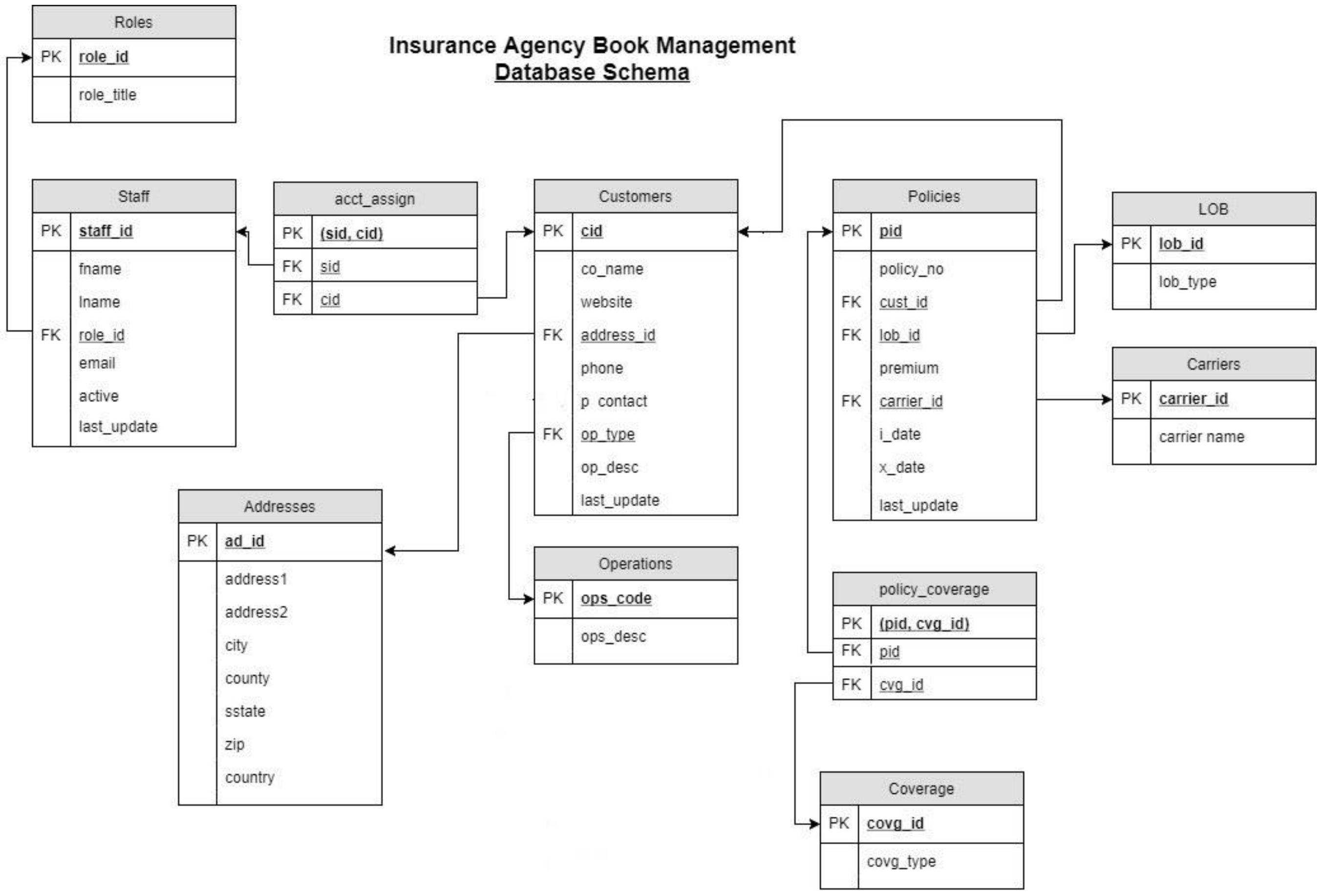
<u>Other secondary relationships include:</u>

- Customers must have a type of business operation, but more than one customer (company) can have the same operations.
- Staff must have one assigned role, but a role can apply to many different staff members.
- Customers must have an address, and an address could apply to more than one customer (Think office building or shared space).
- Policies have are one line of business (type), but a line of business can apply to more than one policy.
-

Final ER Diagram



Final Schema



Data Definition Queries

```
See attached files for details.

DROP TABLE IF EXISTS `LOB`;
DROP TABLE IF EXISTS `Roles`;
DROP TABLE IF EXISTS `Operations`;
DROP TABLE IF EXISTS `Carriers`;
DROP TABLE IF EXISTS `Coverage`;
DROP TABLE IF EXISTS `Addresses`;
DROP TABLE IF EXISTS `Contacts`;
DROP TABLE IF EXISTS `Policies`;
DROP TABLE IF EXISTS `Staff`;
DROP TABLE IF EXISTS `Customers`;
DROP TABLE IF EXISTS `Acct_Assign`;
DROP TABLE IF EXISTS `Policy_Coverage`;


CREATE TABLE LOB (
      lob_id     SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT,
      lob_type VARCHAR(25) NOT NULL,
      PRIMARY KEY (lob_id)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8;


CREATE TABLE Roles (
      role_id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT,
      role_title VARCHAR(25) NOT NULL,
      PRIMARY KEY (role_id)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8;


CREATE TABLE Operations (
      ops_code INT(11) UNSIGNED NOT NULL,
      ops_desc VARCHAR(50) NOT NULL,
      PRIMARY KEY (ops_code)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8;


CREATE TABLE Carriers (
      carrier_id INT(11) UNSIGNED NOT NULL AUTO_INCREMENT,
      carrier_name VARCHAR(35) NOT NULL,
      PRIMARY KEY (carrier_id)
)ENGINE=InnoDB  DEFAULT CHARSET=utf8;
```

```
CREATE TABLE Coverage (
     covg_id INT(11) NOT NULL AUTO_INCREMENT,
     covg_type VARCHAR(100) NOT NULL,
     PRIMARY KEY (covg_id)
)ENGINE=InnoDB  DEFAULT CHARSET=utf8;


CREATE TABLE Addresses (
     ad_id INT(11) UNSIGNED NOT NULL AUTO_INCREMENT,
     address1 VARCHAR(50) NOT NULL,
     address2 VARCHAR(50) DEFAULT NULL,
     city VARCHAR(50) NOT NULL,
     county VARCHAR(50) DEFAULT NULL,
     sstate VARCHAR(3) NOT NULL,
     zip  VARCHAR(15) DEFAULT NULL,
     country   VARCHAR(30) NOT NULL,
     PRIMARY KEY (ad_id)
)ENGINE=InnoDB  DEFAULT CHARSET=utf8;



CREATE TABLE Customers (
     cid INT(11) UNSIGNED NOT NULL AUTO_INCREMENT,
     co_name   VARCHAR(50) NOT NULL,
     website VARCHAR(100) DEFAULT NULL,
     address_id INT(11) UNSIGNED NOT NULL,
     phone VARCHAR(20) NOT NULL,
     p_contact VARCHAR(35),
     op_type   INT(11) UNSIGNED,
     op_desc TEXT,
     last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON
UPDATE CURRENT_TIMESTAMP,
     PRIMARY KEY(cid),
     KEY idx_co_name (co_name),
     CONSTRAINT fk_address_id FOREIGN KEY (address_id) REFERENCES
Addresses (ad_id) ON DELETE RESTRICT ON UPDATE CASCADE,
     CONSTRAINT fk_op_type FOREIGN KEY (op_type) REFERENCES Operations
(ops_code) ON DELETE RESTRICT ON UPDATE CASCADE
)ENGINE=InnoDB  DEFAULT CHARSET=utf8


CREATE TABLE Staff (
     staff_id INT(11) UNSIGNED NOT NULL AUTO_INCREMENT,
     fname VARCHAR(50) NOT NULL,
     lname VARCHAR(50) NOT NULL,
     role_id SMALLINT UNSIGNED NOT NULL,
     email VARCHAR(50) DEFAULT NULL,
     active BOOLEAN NOT NULL DEFAULT TRUE,
	last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON
UPDATE CURRENT_TIMESTAMP,
	PRIMARY KEY (staff_id),
	CONSTRAINT fk_role_id FOREIGN KEY (role_id) REFERENCES Roles
(role_id) ON DELETE RESTRICT ON UPDATE CASCADE
)ENGINE=InnoDB  DEFAULT CHARSET=utf8;


CREATE TABLE Acct_Assign (
	cid INT(11) UNSIGNED NOT NULL,
	sid INT(11) UNSIGNED NOT NULL,
	PRIMARY KEY (cid, aid),
	CONSTRAINT fk_ac_cid FOREIGN KEY (cid) REFERENCES Customers (cid)
ON DELETE CASCADE ON UPDATE CASCADE,
	CONSTRAINT fk_ac_aid FOREIGN KEY (aid) REFERENCES Staff
(staff_id) ON DELETE RESTRICT ON UPDATE CASCADE
)ENGINE=InnoDB  DEFAULT CHARSET=utf8;



CREATE TABLE Policies (
	pid INT(11) UNSIGNED NOT NULL AUTO_INCREMENT,
	policy_no VARCHAR(20) NOT NULL,
	acct_id INT(11) UNSIGNED NOT NULL,
	lob_id SMALLINT UNSIGNED NOT NULL,
	premium DECIMAL(10, 2) NOT NULL,
	carrier_id INT(11) UNSIGNED NOT NULL,
	status VARCHAR(25) NOT NULL,
	i_date DATE NOT NULL,
	x_date DATE NOT NULL,
	last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON
UPDATE CURRENT_TIMESTAMP,
	PRIMARY KEY (pid),
	CONSTRAINT unique_p UNIQUE (policy_no, i_date),
	CONSTRAINT fk_acct_id_p FOREIGN KEY (acct_id) REFERENCES
Customers (cid) ON DELETE RESTRICT ON UPDATE CASCADE,
	CONSTRAINT fk_p_lob_id FOREIGN KEY (lob_id) REFERENCES LOB
(lob_id) ON DELETE RESTRICT ON UPDATE CASCADE,
	CONSTRAINT fk_carrier_id FOREIGN KEY (carrier_id) REFERENCES
Carriers (carrier_id) ON DELETE RESTRICT ON UPDATE CASCADE
)ENGINE=InnoDB  DEFAULT CHARSET=utf8;




CREATE TABLE Policy_Coverage (
	pid INT(11) UNSIGNED,
	cvg_id INT(11),
     PRIMARY KEY (pid, cvg_id),
     CONSTRAINT fk_pc_pid FOREIGN KEY (pid) REFERENCES Policies (pid)
ON DELETE CASCADE ON UPDATE CASCADE,
     CONSTRAINT fk_pc_cvg FOREIGN KEY (cvg_id) REFERENCES Coverage
(covg_id) ON DELETE CASCADE ON UPDATE CASCADE
)ENGINE=InnoDB  DEFAULT CHARSET=utf8;
```

SAMPLE DATA POPULATION QUERIES

```
INSERT INTO lob (lob_type) VALUES ("Package"), ("Umbrella"), ("Work
Comp"), ("Auto"), ("International"), ("Mono-Prop"), ("Mono-GL"),
("Product Liab."), ("K&R"), ("Exec. Risk");

INSERT INTO Roles (role_title) VALUES ("Producer"), ("Account
Manager"), ("Billing"), ("Claims Manager"), ("Assistant"),
("Underwriter"), ("Loss Control");

INSERT INTO Operations VALUES (00001, "Agriculture"), (00010,
"Mining"), (00015, "Construction"), (00020, "Manufacturing"), (00040,
"Transportation & Communications"), (00050, "Wholesale Trade"),
(00052, "Retail Trade"), (00060, "Finance, Insurance & RealEstate"),
(00070, "Services"), (00091, "Public Administration");

INSERT INTO Carriers (carrier_name) VALUES INSERT INTO carriers
(carrier_name) VALUES ("Liberty Mutual"), ("Chubb"), ("AIG"), ("EMC"),
("Travelers"), ("Farmers"), ("Hanover"), ("Progressive"), ("Zurich"),
("Ace"), ("CNA"), ("SAIF"), ("Safeco"), ("Aetna"), ("Kaiser"),
("Allstate"), ("Allied"), ("Nationwide"), ("Mutual of Omaha"),
("QBE"), ("The Standard"),("Hartford"), ("GEICO"), ("Hancock");

INSERT INTO Coverage (covg_type) VALUES ("Building"), ("Personal
Property"), ("Equipment"), ("Products Completed Operations"),
("Premise Liability"), ("Hired Non Owned"), ("Excess"), ("Products"),
("Crops"), ("Int'l Travel"), ("Storm"), ("Named Risk"), ("Liquor
Liab"), ("Property Damage"), ("Bodily Injury"), ("All Risk"), ("Stated
Risk"), ("Fraud"), ("Theft"), ("Errors & Omission "), ("Group Health
Stuff - placeholder"),("Valuables");

INSERT INTO Addresses (address1, address2, city, county, sstate, zip,
country) VALUES ("OSU Example 1","104 Kerr Admin Bldg # 1011",
"Corvallis", "Benton", "OR", "97331", "USA"), ("OSU Example 2","104
Kerr Admin Bldg # 1011", "Corvallis", "Benton", "OR", "97331", "USA"),
("OSU Example 3","104 Kerr Admin Bldg # 1011", "Corvallis", "Benton",
"OR", "97331", "USA"), ("OSU Example 4","104 Kerr Admin Bldg # 1011",
"Corvallis", "Benton", "OR", "97331", "USA"), ("OSU Example 5","104
Kerr Admin Bldg # 1011", "Corvallis", "Benton", "OR", "97331", "USA");

INSERT INTO `customers` (`cid`, `co_name`, `website`, `address_id`,
`phone`, `p_contact`, `op_type`, `op_desc`, `last_update`) VALUES
(NULL, 'ABC Company', 'abc.com', '1', '111-111-1111', NULL, '70',
'This company does something', CURRENT_TIMESTAMP), (NULL, 'XYZ
Company', 'xyz.com', '2', '503-111-1111', NULL, '15', 'This company
does nothing...', CURRENT_TIMESTAMP);
```