REQUIREMENTS

1. Primarily system has two users e.i. Vendor and Customer. For a vendor system stores name, email, address, phone, country, zip, skills, profile, cnic and district. For a Customer system stores first name, last name, email, address, phone, country, zip, city and town. Two types of Vendors. They have same information but Vendor Manager manages a Vendor Admin. Vendors can upload products. For a product system stores information like title, description, summary, regular price, sales price, sku, stock, product feature image and product gallery and date of product created. A product must belong to multiple categories. A product might be available in more than one size. A vendor may or may not have a logo (an image) and a vendor must have skills and a vendor must have skill and a vendor may have more than one skill. Where a vendor skill is basically a category of product. But it doesn’t imply that a vendor cannot upload a product which doesn’t belong to her skills (categories). Vendor can be searched or categorized based on district or skills whereas products can be categorized by their category. A customer can place order. For an order system contains order id, date of order placed, status, products, their prices, discount and total (gross) price. For an order system also stores its shipping details in which shipping address, city, state, zip, delivery date and shipping method is stored. Whereas for payment, payment method and details can be stored. System stores user notification (messages for system’s users e.i customer and vendor). For a notification system contains notification id, title, summary, description, link and date. For better integrity of the system, system stores user logs. User logs are basically action’s history performed by users. For logs system sore log id, date, action. A log is related to a particular user. For the flexibility of the system, the system must have a parameter table which contains any extra parameter related to vendor, customer, order a product or any other entity’s special parameter. These parameters must be integrated in the system in order to reduce the rigidity of the system, so if few parameters are added then there’s no need to change the schema and new parameters can be added directly. A customer give feedback about a vendor or a product in the form of rating and comments. Both can be separately given to vendors or products individually.

Noun Analysis – Extracting Entities

As a first step a noun analysis is performed and all unique nouns are were listed down, so we can check one by one which noun represents the entity of the system. For a noun to be an entity of the system, the entity must belong to a universe of discourse and it must have multiple existence. An extra check may perform to evaluate that either the noun is real world entity or not. By keeping these things in mind, we can choose system entities.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Noun | Multiple existence | Real world entity | Concerned/ related | Selected | Description |
| USER | ✓ | ✓ | ✓ | 1 |  |
| CUSTOMER | ✓ | ✓ | ✓ | 1 |  |
| VENDOR | ✓ | ✓ | ✓ | 1 |  |
| PRODUCT | ✗ | ✓ | ✓ | 1 |  |
| SYSTEM | ✗ | ✓ | ✗ | 0 | No need. System itself isn’t an entity. |
| CATEGORY | ✓ | ✗ | ✓ | 1 |  |
| SIZE | ✓ | ✗ | ✓ | 1 |  |
| LOGO | ✓ | ✓ | ✓ | 0 | Image is selected instead. |
| IMAGE | ✓ | ✓ | ✓ | 1 |  |
| SKILL | ✓ | ✗ | ✓ | 1 |  |
| DISTRICT | ✓ | ✓ | ✓ | 1 |  |
| ORDER | ✓ | ✓ | ✓ | 1 |  |
| SHIPPING | ✓ | ✗ | ✓ | 1 |  |
| PAYMENT | ✓ | ✗ | ✓ | 1 |  |
| NOTIFICATION | ✓ | ✗ | ✓ | 1 |  |
| LOG | ✓ | ✗ | ✓ | 1 |  |
| HISTORY | ✓ | ✗ | ✓ | 0 | Already selected as LOG |
| PARAMTER | ✓ | ✗ | ✓ | 1 |  |
| SCHEMA | ✗ | ✗ | ✗ | 0 | Schema isn’t an entity for schema. |
| FEEDBACK | ✓ | ✗ | ✓ | 1 |  |
| DISCOUNT | ✓ | ✗ | ✓ | 1 |  |
|  |  |  |  |  |  |

Adjective Analysis – Extracting Attributes

While entities of the system are defined, then there’s time to know the attributes or properties of these entities to see their behavior. Properties of the entity are the adjective in requirement elicitation which define the noun, or the noun means entity, is based on those adjectives. And hence to extract attributes of the entities an adjective analysis is performed.

|  |  |
| --- | --- |
|  |  |
| Email | Vendor, Customer |
| Type | Vendor, Customer |
| Address | Vendor, Customer, Shipping |
| Phone | Vendor, Customer |
| Country | Vendor, Customer |
| Zip | Vendor, Customer, Shipping |
| Name | Vendor |
| Date | Vendor, Customer, Notification, Log, Product, Shipping |
| Skills | Vendor |
| Profile | Vendor |
| CNIC | Vendor |
| District | Vendor, District |
| First name | Customer |
| Last name | Customer |
| City | Customer, Shipping |
| Town | Customer |
| Title | Product, Notification |
| Description | Product, Notification |
| Summary | Product, Notification |
| Regular price | Product |
| Sales price | Product |
| Sku | Product |
| Stock | Product |
| Category | Product, Category |
| Size | Product, Size |
| Image | Product, Vendor, Image |
| Status | Order |
| Gross | Order |
| Quantity | Order |
| Discount | Order, Discount |
| State | Shipping |
| Delivery time | Shipping |
| Log | Log |
| Action type | Log |
| Parameter | Parameter |
| Comment | Feedback |
| Rating | Feedback |
| Link | Notification |

Verb Analysis – Extracting Relations

In order to perform more accurate analysis and putting them into work further analysis is performed. In this section verbs are extracted from requirement scenario and those verbs represents relation between two entities. While performing a verb noun analysis a verb or a relation is identified by its’ concern with both entities. The entities belonging to that relation might be a ‘subject’ and an ‘object’.

|  |  |  |  |
| --- | --- | --- | --- |
| Relation | Entity 1 | Entity 2 | In words |
| Uploads | Vendor | Product | Vendor uploads Product |
| Place | Customer | Order | Customer place Order |
| Contains | Order | Product | Order contains Products |
| Has | Product, Order | Discount | Product as Discount |
| Receives | User | Notification | User receives notifications |
| Registers | User | Log | User registers log |
| ISA | Vendor, Customer | User | Customer is a User |

RELATIONS

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | USER | CUSTOMER | VENDOR | PRODUCT | OREDER |
|  | User\_id | Customer\_id | Vendor\_id | Product\_id | Order\_id |
|  | Email | Fist\_name | Full\_name | Title | Date |
|  | Password | Last\_name | Slug | Description | Status |
|  | Type | City | Skills | Summary | Gross |
|  | Date | town | Profile | Regular\_price | Shipping\_address |
|  | Address |  | CNIC | Sales\_price | City |
|  | Phone |  | Role | Sku | State |
|  | Country |  | District | stock | zip |
|  | zip |  |  | Size |  |
|  |  |  |  | Category |  |
|  |  |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | DISCOUNT | RATING | NOTIFICATION | LOG | PARAM |
|  | Discount\_id | Rating\_id | Notification\_id | Log\_id | Param\_id |
|  | Value | Rating | Title | Date | Type |
|  | Unit | Approved | Summary | Action\_type | Param |
|  | Date | type | Description | Description | value |
|  | Valid\_until |  | Link |  |  |
|  | Min\_order\_val |  | Date |  |  |
|  | Max\_discount |  |  |  |  |
|  | type |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

RELATION DETAILS

USER:

1. Contains data of every system of users.
2. Contains parameters which are common among VENDOR and CUSTOMER.
3. Contains a parameter ‘type’ which can determine the type of USER e.g. let’s a only a user\_id is provided as a key attribute. Then the ‘type’ can directly reflect that which kind of user it belongs to.
4. As can be seen, it’s a super type and it’s more general and share parameters amont it’s child e.i. VENDOR and CUSTOMER and they’re special cases of USER entity.
5. Also contains login credentials of USER.

CUSTOMER:

1. Contains information of CUSTOMER USER.
2. CUSTOMER is a sub type of more general entity, that is USER. As it holds ‘CUSTOMER ISA USER’ and hence implies the generalization among them. And CUSTOMER will inherit some attributes from USER.

VENDOR:

1. Contains data of VENDOR USER.
2. Sub type of USER and hence inherits its data directly from USER table.
3. VENDOR is logically divided into two types.
   1. VENDOR admin
   2. VENDOR manager
4. VENDOR manager manages another instance of the VENDOR and logically it’s called as VENDOR admin.
5. A special attribute is role which logically categorize VENDOR into three groups.
   1. Pending VENDOR
   2. Admin VENDOR
   3. Manager VENDOR
6. VENDOR entity has a recursive relation with its own entity and hence an attribute named ‘admin\_id’ refers to the instance of the same entity if a VENDOR manager is managing a VENDOR admin. In this case ‘admin\_id’ is pointing towards that VENDOR.
7. If the attribute ‘role’ of VENDOR Is ‘admin’ then the attribute ‘admin\_id’ will be empty or filled with his own primary key. And if it’s empty or doesn’t have its own key it means this instance of VENDOR is manager and it can be confirmed from the role attribute hon.

PRODUCT:

1. Contains instances of PRODUCT that are displayed on the portal.
2. PRODUCT is the central business entity and can be counted as an asset of business.
3. In order to determine future business strategies BUSINESS entity could be used as an analyzing entity in Star or Galaxy schema.

LOG:

1. A special table which is used to store historical data about actions of users.
2. By historical records means data of this table must be read only and records can be added but updating and deletion from this table must not be allowed to user.
3. LOG keep records of USER actions. If user did something that must be tracked, then this table can help to look for the actor of the particular action.
4. This table can also be used to get total population of the table even if some records have been deleted.

PARAM:

1. A special Relational table which may used to accommodate minor changes in metadata abruptly.
2. If an attribute wasn’t added or didn’t mentioned in initial requirements, then a minor change in metadata can be accommodate immediately without updating the database schema.
3. For any entity a separate PARAM table can be created or can be managed inside the table.
4. As for all ids of the system, all entities have same data type of id and range so they can be accommodated in just one column.
5. An attribute ‘type’ can be sued to distinguish for which entity this id belongs to.

RATING:

1. CUSTOMER can give rating to a VENDOR or to a PRODUCT individually.
2. An attribute ‘type’ will determine about the entity this RATING belongs to e.i. either the RATING is given to a VENDOR or to a PRODUCT.
3. A special attribute of ‘approved’ is added so an Admin can view the given feedback before allowing this RATING to show publicly.
4. An ‘id’ attribute might belong to a VENDOR or to a PRODUCT.

ORDER:

1. When a CUSTOMER place an order then instance of this order is placed inside ORDER entity and related attributes are stored accordingly.
2. ORDER contains record of CUSTOMER who placed this order and details about shipping.
3. ORDER also has information about which SHIPMENT method will be used and which PAYMENT method is linked.
4. An ORDER may also have DISCOUNT.

DISCOUNT:

1. All types of DISCOUNT are placed there. Each of which has different kind.
2. DISCOUNT can be given to PRODUCT individually or it can be given on the placement of ORDER as a collective.
3. The gross total of an order is calculated based on all calculations which are separately stored in different tables.

NORMALIZATION

1NF:

1. All entities are given to a unique identifier which are generated irrespective of the other attributes of the entity and identify each instance uniquely. So by default all entities are in 1NF form.

2NF: Violation of 2nd NF occurs in many entities because non key attributes in many entities identify other non key attributes of the same entity.

1. The skills of VENDOR which is a non key attribute, further define other instances of VENDOR, and hence they are separated from VENDOR entity.
2. Products in PRODUCT table also has a non key attributes which define other non key attributes and points to different instances of the same entity. And hence SIZE and CATEGORY are also separated from PRODUCT entity and both of them has their own Primary Key to identify each instance of SIZE and CATEGORY uniquely.
3. District was also a non key attribute which was used to identity other non key attributes of VENDOR entity. And so DISTRICT entity must be created separated from VENDOR.
4. Type is a non key attribute but it filters out the result of other instances of the entity like USER, LOG and DISCOUNT. So it must be a sperate entity so we can link each entity with its respected type and then can filter out based on their types.
5. Status represents the state of many activities and hence it can be used to identify a set of other instances of the class so the STATUS have been put out from entities like USER, ORDER and NOTIFICATION and placed as a separate entity.
6. Role is a special attribute in USER entity and it determines weather the current user is VENDOR as admin or VENDOR as a manager behave. So it logically divide the VENDOR entity into three groups. The other one is pending VENDOR. So in that sense it must also treated as separately.

Note. Zip attribute is also a non key attribute and determines the other instances of entity but we do not put it as a separate entity because it might change it’s value and we might not know which values should be placed in the table. And second reason is that, this table is filled by USER of the system instead of Admin so it have been accommodated inside their tables.

3NF: identification of associative dependency and removing it from inside the tables are performed in this step.

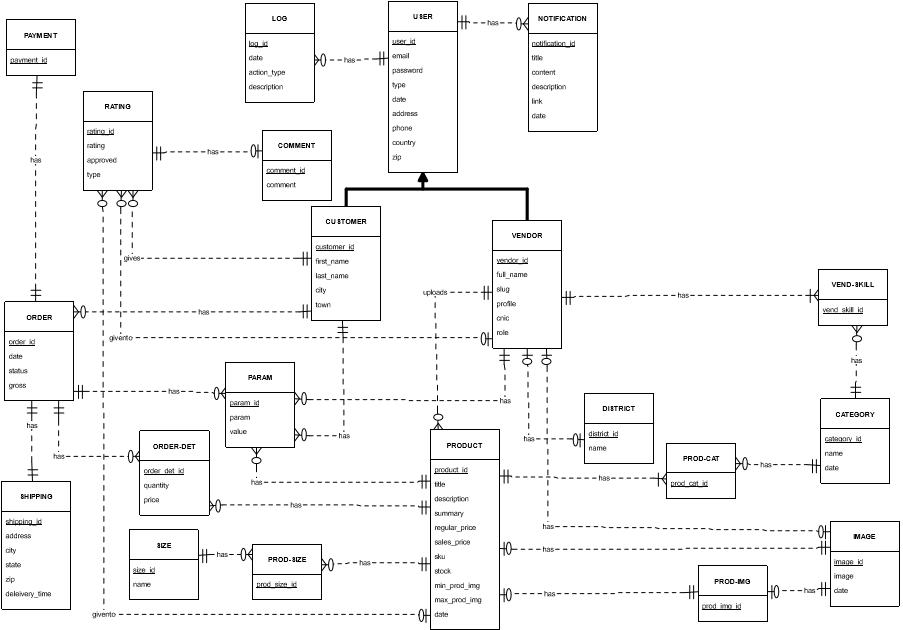
1. Payment and shipment method in ORDER are separated from ORDER because Payment details depend on payment\_id which is a key attribute and it depends upon order\_id which is another key attribute and hence they exhibit associative dependency. To remove this dependency PAYMENT and SHIPMENT entities are created.

BCNF: All entities are now in BCNF and no need to execute it for entities of the system because 3NF produced the same results.

4NF: to reduce dependencies and anomalies the many to many relationship between entities are decomposed with the help of junction tables.

1. When the relation between ORDER nad PRODUCT is decomposed then a new entity is created whos name is ORDER-DETAIL which includes each item in order and contains product\_id, its quantity, price and discount.
2. Relation between PRODUCT and CATEGORY is also decomposed to PRODUCT-CATEGOTY which binds together all categories related to a particular PRODUCT.
3. A junction table among PRODUCT and SIZEE is also decomposed to form PROD-SIZE which contains different sizes of available for a PRODUCT.
4. As it is stated, a PRODUCT has different images placed in a single table and to extract images belongs to PRODUCT are placed in PROD-IMG entity.
5. VENDOR may have multiple skills in CATEGORY and one CATEGORY may belongs to many VENDOR. By keeping these things in mind a junction table is created as VEND-SKILLS which clusters VENDOR skill at one place.

DIAGRAM:



SCHEMA

1. COMMON\_USER

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | User\_id | Int (6) | PK, NN |
|  | Email | Varchar (30) | U, NN |
|  | Password | Varchar (30) | NN |
|  | Type | Varchar (20) | FK |
|  | Date | Date |  |
|  | Address | Varchar (100) |  |
|  | Phone | Varchar (15) | U |
|  | Country | Varchar (20) |  |
|  | zip | Int (5) |  |
|  | Status | Int (6) | FK |

1. TYPE

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Type\_id | Int (6) | PK |
|  | Name | Varchar (30) | U, NN |
|  | Description | Varchar (100) |  |

1. NOTIFICATION

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Notification\_id | Int (6) | PK |
|  | Title | Varchar (30) | NN |
|  | Summary | Varchar (100) |  |
|  | Description | Varchar (500) |  |
|  | Link | Varchar (120) |  |
|  | date | Date | NN |
|  | User\_id | Int (6) | FK |
|  | Status\_id | Int (6) | FK |

1. LOG

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Log\_id | Int (6) | PK |
|  | Type\_id | Int (6) | FK |
|  | Date | Date | NN |
|  | Description | Varchar (120) |  |
|  | User\_id | Int (6) | FK |

1. VENDOR

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Vendor\_id | Int (6) | PK |
|  | Full\_name | Varchar (40) | U, NN |
|  | Slug | Varchar (50) | U |
|  | Profile | Varchar (1000) |  |
|  | CNIC | Varchar (15) | U, NN |
|  | Role\_id | Int (6) | FK |
|  | User\_id | Int (6) | FK, NN |
|  | Image\_id | Int (6) | FK |
|  | District\_id | Int (6) | FK |
|  | Admin\_id | Int (6) | FK |

1. ROLE

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Role\_id | Int (6) | PK |
|  | Name | Varchar (30) | U, NN |
|  | Description | Varchar (100) |  |

1. CUSTOMER

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Customer\_id | Int (6) | PK |
|  | First\_name | Varchar (20) |  |
|  | Last\_name | Varchar (20) |  |
|  | City | Varchar (20) | Non key attribute |
|  | Town | Varchar (20) | Non key attribute |
|  | User\_id | Int (6) | FK, NN |

1. PRODUCT

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Product\_id | Int (6) | PK |
|  | Title | Varchar (30) | NN |
|  | Description | Varchar (1000) |  |
|  | Summary | Varchar (500) |  |
|  | Regular\_price | Int (6) |  |
|  | Sales\_price | Int (6) |  |
|  | Sku | Varchar (6) | U |
|  | Stock | Int (3) |  |
|  | Min\_prod\_img | Int (1) |  |
|  | Max\_prod\_img | Int (1) |  |
|  | date | Date |  |
|  | Image\_id | Int (6) | FK |
|  | Discount\_id | Int (6) | FK |
|  | Status \_id | Int (6) | FK |
|  | Vendor\_id | Int (6) | FK, NN |

1. DISTRICT

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | District\_id | Int (6) | PK |
|  | Name | Varchar (30) | U, NN |
|  | Description | Varchar (100) |  |

1. CATEGORY

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Category\_id | Int (6) | PK |
|  | Name | Varchar (30) | U, NN |
|  | Description | Varchar (100) |  |

1. SIZE

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Size\_id | Int (6) | PK |
|  | Name | Varchar (20) | U, NN |
|  | Description | Varchar (100) |  |

1. ORDERS

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Order\_id | Int (6) | PK |
|  | Date | Date | NN |
|  | Gross | Int (6) |  |
|  | Shipping\_address | Varchar (150) |  |
|  | City | Varchar (20) | Non key attribute |
|  | State | Varchar (20) | Non key attribute |
|  | zip | Int (5) | Non key attribute |
|  | Status\_id | Int (6) | FK |
|  | Customer\_id | Int (6) | FK |
|  | Payment\_id | Int (6) | FK |
|  | Shipment\_id | Int (6) | FK |
|  | Discount\_id | Int (6) | FK |

1. STATUS

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Status\_id | Int (6) | PK |
|  | Name | Varchar (30) | U, NN |
|  | Description | Varchar (100) |  |

1. DISCOUNT

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Discount\_id | Int (6) | PK |
|  | Name | Varchar (30) | U, NN |
|  | Type\_id | Int (6) | FK |
|  | Unit | Float (4, 1) |  |
|  | Date | Date | NN |
|  | Valid\_until | Date |  |
|  | Min\_order\_val | Int (6, 1) |  |
|  | Max\_disc\_val | Float (6, 1) |  |

1. IMAGE

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Image\_id | Int (6) | PK |
|  | Image | Int (10) | U |
|  | Date | Date |  |

1. PROD\_IMG

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Prod\_img\_id | Int (6) | PK |
|  | Image\_id | Int (6) | FK |
|  | Product\_id | Int (6) | FK |

1. PAYMENT

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Payment\_id | Int (6) | PK |
|  | Method | Varchar (30) | U, NN |
|  | Description | Varchar (100) |  |

1. SHIPMENT

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Shipment\_id | Int (6) | PK |
|  | Method | Varchar (30) | U, NN |
|  | Description | Varchar (100) |  |

1. RATING

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Rating\_id | Int (6) | PK |
|  | Rating | Int (1) | NN |
|  | Approved | Int (1) |  |
|  | Type | Varchar (20) |  |
|  | Customer\_id | Int (6) | FK |

1. COMMENT

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Comment\_id | Int (6) | PK |
|  | Comment | Varchar (200) | NN |
|  | Rating\_id | Int (6) | FK |

1. PROD\_RATING

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Prod\_rating\_id | Int (6) | PK |
|  | Rating\_id | Int (6) | FK |
|  | Product\_id | Int (6) | FK |

1. VEND\_RATING

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Vend\_rating\_id | Int (6) | PK |
|  | Rating\_id | Int (6) | FK |
|  | Vendor\_id | Int (6) | FK |

1. PARAM

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Param\_id | Int (6) | PK |
|  | Type | Varchar (20) |  |
|  | Param | Varchar (30) | NN |
|  | Value | Varchar (120) |  |

1. CUST\_PARAM

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Cust\_param\_id | Int (6) | PK |
|  | Param\_id | Int (6) | FK |
|  | Customer\_id | Int (6) | FK |

1. ORDER\_PARAM

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Order\_param\_id | Int (6) | PK |
|  | Param\_id | Int (6) | FK |
|  | Order\_id | Int (6) | FK |

1. PROD\_PARAM

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Order\_param\_id | Int (6) | PK |
|  | Param\_id | Int (6) | FK |
|  | Product\_id | Int (6) | FK |

1. VEND\_PARAM

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Vend\_param\_id | Int (6) | PK |
|  | Param\_id | Int (6) | FK |
|  | Vendor\_id | Int (6) | FK |

1. ORDER\_DET

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Order\_det\_id | Int | PK |
|  | Quantity | Int |  |
|  | Price | Int |  |
|  | Product\_id | Int | FK |
|  | Order\_id | int | FK |

1. PROD\_SIZE

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Prod\_size\_id | Int | PK |
|  | Product\_id | Int | FK |
|  | size\_id | int | FK |

1. VEND\_SKILL

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Vend\_skill\_id | Int | PK |
|  | Vendor\_id | Int | FK |
|  | Category\_id | int | FK |

1. PROD\_CAT

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Prod\_cat\_id | Int | PK |
|  | Product\_id | Int | FK |
|  | Category\_id | int | FK |