

MAGIK SNOWBOARDS INC.



Database Architecture Specification

Table of Contents:

Executive Summary:	3
Logical Database Architecture:	4
First Class Table Descriptions:	5
Customers Table	5
CustomerOrders Table	6
DeptCodes Table	7
Employees Table	8
Events Table	9
People Table	10
Products Table	11
ProRiders Table	13
PTypes Table	14
SupplyOrders Table	15
Vendors Table	16
ZipCodes Table	17
Associative Entities and Entity Subtypes:	18
Accessories Table	18
Competes Table	19
CustomerLineItems Table	20
Services Table	21
Snowboards Table	22
SupplyLineItems Table	23
Reports:	24
Promotional Mailing Report	24
Snowboard Inventory Report	24
Accessories Inventory Report	25
Views:	26
Inventory View	26
Employee View	26
Security:	28
Administrator role:	28
User Role:	29
Looking Forward:	31

Executive Summary:

This document details the database architecture and specification for the MagiK Snowboards Inc. company. Each section will take the reader through the database from different aspects and varying levels of detail.

The first section Logical Database Architecture shows the layout of the database and the interrelations between each piece of the business. The customer relational, human resources, vendor supply, sales, and event organization functions are all contained and explained herein.

The next section of the document explains the detail behind each first class object (e.g. People, Employees, Events, Vendors, etc...). The functional dependencies, the Structured Query Language (SQL) CREATE statements, and a sample (snap shot) of the data in the table described are shown.

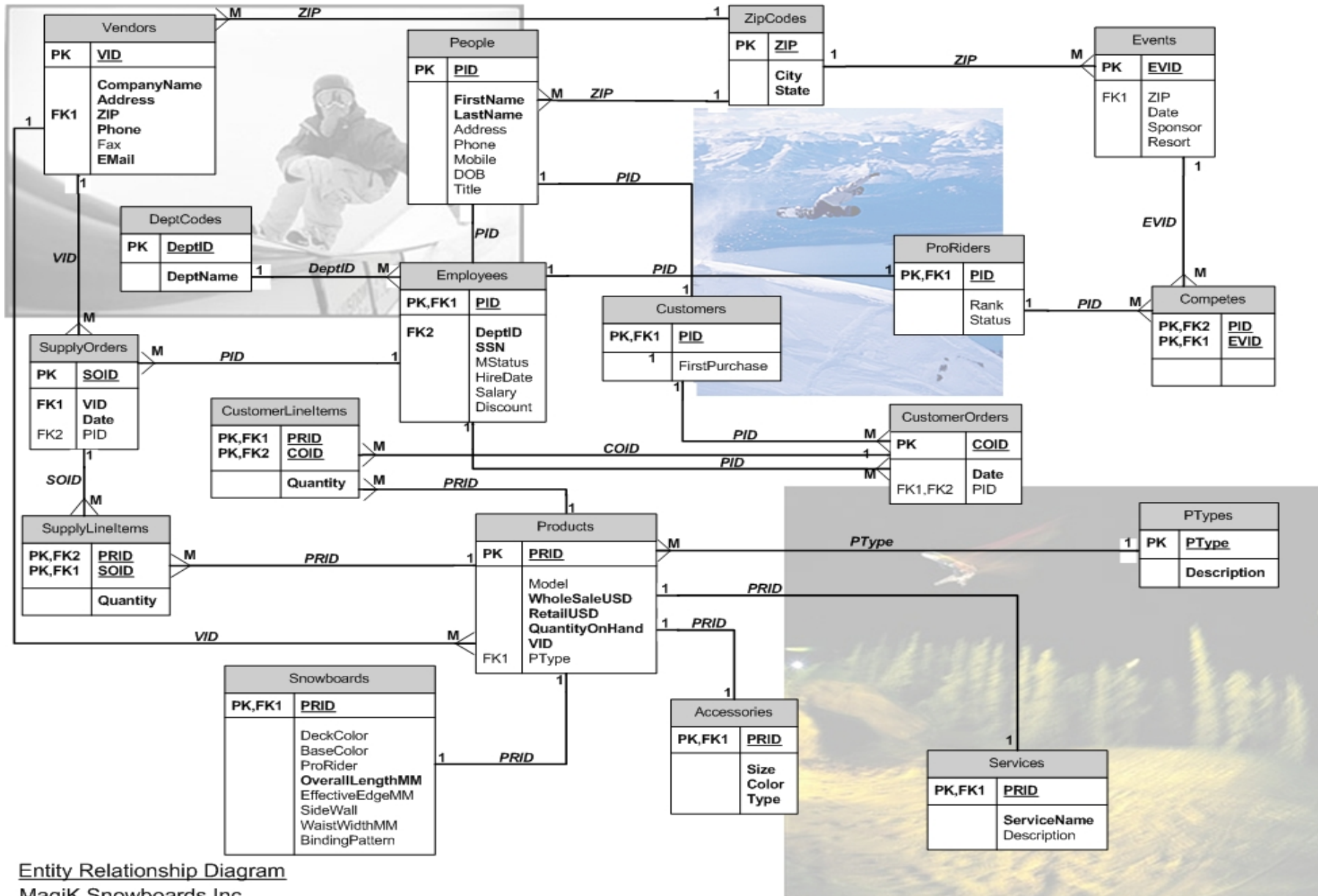
The third section describes the associative entities used to demonstrate the many-to-many relationships implemented between the first class objects as well as the entity sub-type objects that have no other entities dependent on them.

Section four and five demonstrate an example the reporting and view capabilities of the database.

The last two sections demonstrate the security granted to each type of user for different portions of the database and the improvements that should be considered going forward.

Logical Database Architecture:

Below is the Entity Relationship Diagram that demonstrates the interactions (relations) between the entities that make up the MagiK Snowboards Inc. Database.



Entity Relationship Diagram
MagiK Snowboards Inc.

First Class Table Descriptions:

This section describes each of the first class entities and first class sub-types shown in the entity relationship diagram (ERD) in the previous section.

Customers Table

The Customers table has one unique field, the FirstPurchase attribute, that allows the calculation for how long a person has been a customer. This table is a sub-type of the people table with the intent of keeping track of all customers that make purchases at MagiK Snowboards.

Functional Dependencies:

PID → FirstPurchase

SQL CREATE statement:

```
CREATE TABLE Customers (  
    PID          char (9) not null,  
    FirstPurchase datetime,  
primary key (PID),  
foreign key (PID) references People (PID)  
)
```

Sample Data:

PID	FirstPurchase
-----	-----
PE0000001	1998-10-05 00:00:00.000
PE0000004	1997-11-06 00:00:00.000
PE0000005	1983-09-05 00:00:00.000
PE0000006	1989-10-31 00:00:00.000
PE0000010	1991-08-06 00:00:00.000

CustomerOrders Table

The CustomerOrders table uses the COID attribute that starts with “COD” followed by a 9 digit hex number. Customer Orders show what customer and which employee were involved in a specific sale.

Functional Dependencies:

COID → EPID, CPID, Date

SQL CREATE statement:

```
CREATE TABLE CustomerOrders (  
    COID      char (12) not null,  
    EPID      char (9)  not null,  
    CPID      char (9)  not null,  
    Date      datetime not null,  
    primary key (COID),  
    foreign key (EPID) references Employees (PID),  
    foreign key (CPID) references Customers (PID)  
)
```

Sample Data:

COID	EPID	CPID	Date
-----	-----	-----	-----
COD000000001	PE0000001	PE0000004	2004-10-24 00:00:00.000
COD000000002	PE0000010	PE0000005	2004-11-05 00:00:00.000
COD000000003	PE0000001	PE0000006	2004-11-24 00:00:00.000

DeptCodes Table

The DeptCodes table is a fixed table that could be used in a pick-list that has department codes for the employees working at MagiK Snowboards.

Functional Dependencies:

DeptID → DeptName

SQL CREATE statement:

```
CREATE TABLE DeptCodes (  
    DeptID    char (5)        not null,  
    DeptName  varchar (20)    not null,  
    primary key (DeptID)  
)
```

Sample Data:

DeptID	DeptName
MS001	Executive
MS002	Accounting
MS003	Purchasing
MS004	Sales
MS005	Pro

Employees Table

The Employees table is a sub-type of the People table. There are specific attributes associated with employees (e.g. salary and SSN) that are stored in the Employees table.

Functional Dependencies:

PID → DeptID, SSN, MStatus, HireDate, SalaryUSD, Discount

SQL CREATE statement:

```
CREATE TABLE Emplpyees (  
    PID          char (9) not null,  
    DeptID       char (5) not null default(MS006),  
    SSN          char (9) not null unique,  
    MStatus      char (1) check (MStatus='S' or MStatus='M' or MStatus='D' or MStatus='W'),  
    HireDate     datetime not null,  
    Salary       money    not null,  
    Discount     numeric (3,2) not null default(0.10),  
primary key (PID),  
foreign key (PID) references People (PID),  
foreign key (DeptID) references DeptCodes (DeptID)  
)
```

Sample Data:

PID	DeptID	SSN	MStatus	HireDate	Salary	Discount
PE0000001	MS004	100628999	M	2000-11-27 00:00:00.000	32000.0000	.15
PE0000002	MS002	118669073	M	2000-11-27 00:00:00.000	80000.0000	.30
PE0000003	MS003	100770007	S	1998-10-07 00:00:00.000	60000.0000	.15
PE0000008	MS005	111111117	D	1996-06-06 00:00:00.000	90000.0000	1.00
PE0000009	MS001	999897999	S	2003-01-01 00:00:00.000	300000.0000	.50
PE0000010	MS004	152639877	M	1999-04-25 00:00:00.000	50000.0000	.20

Events Table

The Events table holds records of competitive events that take place at different resorts. Pro Riders can compete in these events.

Functional Dependencies:

EVID → ZIP, Date, Sponsor, Resort

SQL CREATE statement:

```
CREATE TABLE Events (  
    EVID      char (7)  not null,  
    ZIP       char (5)  not null,  
    Sponsor   varchar (30)  not null default ('MagiK Snowboards Inc.'),  
    Resort    varchar (20)  not null default ('Hunter Mountain'),  
    Date      datetime  not null,  
    primary key (EVID),  
    foreign key (ZIP) references ZipCodes (ZIP)  
)
```

Sample Data:

EVID	ZIP	Sponsor	Resort	Date
EV00001	12442	Budweiser	Hunter Mountain	2002-12-15 00:00:00.000
EV00002	12442	Burton	Hunter Mountain	2003-01-18 00:00:00.000
EV00003	05751	Coors	Killington	2003-03-15 00:00:00.000
EV00004	12442	Volcom	Hunter Mountain	2003-04-15 00:00:00.000

People Table

The People table is the master table for all people that either work (Employees), purchase (Customers), or ride professionally (ProRiders) at MagiK Snowboards. The PID is the primary key for this and all its entity sup-types.

Functional Dependencies:

PID → FirstName, LastName, Title, Address, ZIP, DOB, Phone, Mobile

SQL CREATE statement:

```
CREATE TABLE People (  
    PID          char (9)          not null,  
    FirstName    varchar (20)      not null,  
    LastName     varchar (20)      not null,  
    Title        varchar (4)       not null check (Title='Mr' or Title='Mrs' or Title='Ms' or Title='Miss'),  
    Address      varchar (30)      not null,  
    ZIP          char (5)          not null,  
    DOB         datetime          not null,  
    Phone       char (10)         not null,  
    Mobile      char (10),  
    primary key (PID),  
    foreign key (ZIP) references ZipCodes (ZIP)  
)
```

Sample Data:

PID	FirstName	LastName	Title	Address	ZIP	DOB	Phone	Mobile
PE0000001	Joshua	Matheus	Mr	172 Wilbur Blvd	12603	1978-02-26 00:00:00.000	8454719099	8456492816
PE0000002	Carolyn	Matheus	Ms	172 Wilbur Blvd	12603	1976-10-19 00:00:00.000	8454719099	9144748310
PE0000003	James	Bond	Mr	007 SpiesLikeUs Ct	12442	1964-07-07 00:00:00.000	5185559099	5189690007
PE0000004	Love	Doctor	Miss	876 NaughtyTime Blvd	12601	1980-01-13 00:00:00.000	8454629089	8456496879
PE0000005	Darth	Vader	Mr	327 DeathStar Drive	12603	1977-05-18 00:00:00.000	8454541138	8456497758
PE0000006	Ben	Dover	Mr	34 Pie Ln	12590	1961-03-25 00:00:00.000	8452973325	914555DUMB
PE0000007	Johnny	Neumonic	Mr	13 Wilbur Blvd	12603	1976-04-24 00:00:00.000	8454712365	8456548796
PE0000008	Jenna	Jameson	Ms	174 Wilbur Blvd	12603	1974-10-31 00:00:00.000	8454716666	914564LOVE
PE0000009	Chewbacca	Matheus	Mr	172 Wilbur Blvd	12603	2002-03-11 00:00:00.000	8454719099	845649PIMP
PE0000010	Big "Hank"	Matheus	Mr	127 Spackenkill Rd	12603	1951-10-28 00:00:00.000	8454626311	9149240492

Products Table

The Products table is the parent entity for snowboards, accessories, and services.

Functional Dependencies:

PRID → Model, WholeSaleUSD, RetailUSD, Vendor, QuantityOnHand, PType

SQL CREATE statement:

```
CREATE TABLE Products (  
    PRID          char (8)          not null,  
    Model         varchar (20),  
    WholeSaleUSD  money,  
    RetailUSD     money,  
    VID           char (7),  
    QuantityOnHand int,  
    PType        char (2) not null default('AC'),  
    primary key (PRID),  
    foreign key (VID) references Vendors (VID),  
    foreign key (PType) references PTypes (PType)  
)
```

Sample Data:

PRID	Model	WholeSaleUSD	RetailUSD	VID	QuantityOnHand	PType
PR000001	Jameson	400.0000	500.0000	V000006	3	SB
PR000002	Powers	410.0000	550.0000	V000001	2	SB
PR000003	Mr Freeze	90.0000	150.0000	V000005	5	JK
PR000004	AK Outland	110.0000	200.0000	V000001	6	PT
PR000005	Ice Man	50.0000	90.0000	V000004	15	GG
PR000006	Blinder	40.0000	80.0000	V000005	12	GG
PR000007	Floater	350.0000	420.0000	V000003	3	SB
PR000008	Big Hands	30.0000	60.0000	V000001	10	GL
PR000009	Slick Willy	5.0000	12.0000	V000004	15	WX

PR000010 Big Foot	120.0000	180.0000	V000002 10	BT
PR000011 Sharpen	NULL	10.0000	NULL NULL	SV
PR000012 Base	NULL	40.0000	NULL NULL	SV
PR000013 Wax	NULL	20.0000	NULL NULL	SV

ProRiders Table

ProRiders is an entity sub-type of People and Employees.

Functional Dependencies:

PID → Rank, Status

SQL CREATE statement:

```
CREATE TABLE ProRiders (  
    PID          char (9)    not null,  
    Rank         numeric (3,0) default (1),  
    Status       varchar (8) not null check (Status = 'Active' or Status = 'Disabled' or Status = 'InActive' or  
Status = 'Retired'),  
    primary key (PID),  
    foreign key (PID) references Employees (PID)  
)
```

Sample Data:

PID	Rank	Status
PE0000008	93	Active

PTypes Table

PTypes was added to the database architecture to allow better query capability on specific types of products.

Functional Dependencies:

PType → Description

SQL CREATE statement:

```
CREATE TABLE PTypes (  
    PType      char (2)    not null default('AC'),  
    Description varchar (30),  
    primary key (PType)  
)
```

Sample Data:

PType Description

```
-----  
BN    Bindings  
BT    Boots  
GG    Goggles  
GL    Gloves  
HM    Helmet  
JK    Jacket  
PT    Pants  
SB    Snow Board  
SV    Services  
WX    Wax
```

SupplyOrders Table

The SupplyOrders table has the orders that are made to each vendor for MagiK Snowboards.

Functional Dependencies:

SOID → VID, PID, Date

SQL CREATE statement:

```
CREATE TABLE SupplyOrders (  
    SOID      char (12)      not null,  
    PID       char (9)       not null,  
    Date      datetime      not null,  
    VID       char (7),  
    primary key (SOID),  
    foreign key (VID) references Vendors (VID),  
    foreign key (PID) references Employees (PID)  
)
```

Sample Data:

SOID	PID	Date	VID
SID000000001	PE0000003	2003-05-25 00:00:00.000	V000001
SID000000002	PE0000003	2003-06-25 00:00:00.000	V000002
SID000000003	PE0000003	2003-07-25 00:00:00.000	V000004
SID000000004	PE0000003	2003-08-25 00:00:00.000	V000006

Vendors Table

The Vendors table holds records of all the vendors that MagiK Snowboards.

Functional Dependencies:

VID → CompanyName, Address, ZIP, Phone, Fax, EMail

SQL CREATE statement:

```
CREATE TABLE Vendors (  
    VID          char (7)          not null,  
    CompanyName varchar (20)       not null,  
    Address      varchar (30)      not null,  
    ZIP          char (5)          not null,  
    Phone       char (9)          not null,  
    Fax         char (9),  
    EMail       varchar (30),  
    primary key (VID),  
    foreign key (ZIP) references ZipCodes (ZIP)  
)
```

Sample Data:

VID	CompanyName	Address	ZIP	EMail	Phone	Fax
V000001	Burton	34 Snow Rd	05751	Jake@Burton.com	4789869669	4789865555
V000002	Volcom	54 Blizzard Dr	05751	Jake@Burton.com	4789869669	4789865555
V000003	Ride	94 Mountain Rd	12442	Bob@Ride.com	5189869669	5189865555
V000004	RobotFood	27 Chilled Rd	12590	Jussi@RF.com	8459869669	8459865555
V000005	Jeenyus	999 Main Street	12603	KevinJones@Jeenyus.com	8459869669	8459865555
V000006	Sims	89 C4 Dr	05751	MarcFrank@Sims.com	5189869669	5189865555

ZipCodes Table

Functional Dependencies:

ZIP → City, State

SQL CREATE statement:

```
CREATE TABLE ZipCodes (  
    ZIP          char (5)          not null,  
    City         varchar (20)      not null,  
    State        char(2)           not null default ('NY'),  
    primary key (ZIP)  
)
```

Sample Data:

ZIP	City	State
05751	Rutland	VT
12442	Hunter	NY
12590	Wappingers Falls	NY
12601	Poughkeepsie	NY
12603	Poughkeepsie	NY

Associative Entities and Entity Subtypes:

Accessories Table

Functional Dependencies:

PRID → Size, Color, Type

SQL CREATE statement:

```
CREATE TABLE Accessories (  
    PRID      char (8)      not null,  
    Size      varchar (10)  not null,  
    Color     char(2)       not null,  
    Type      varchar (30),  
    primary key (PRID),  
    foreign key (PRID) references Products (PRID)  
)
```

Sample Data:

PRID	Size	Color	Type
PR000003	M	BL	Shell
PR000004	L	GR	Down
PR000005	L	GN	Single Lens
PR000006	M	BK	Double Lens
PR000008	S	RD	Five Finger Glove
PR000009	L	PK	All weather wax
PR000010	8	OG	Soft Boot

(7 row(s) affected)

Competes Table

Functional Dependencies:

EVID, PID →

SQL CREATE statement:

```
CREATE TABLE Competes (  
    EVID      char (7) not null,  
    PID      char (9) not null,  
    primary key (EVID, PID),  
    foreign key (EVID) references Events (EVID),  
    foreign key (PID) references ProRiders (PID)  
)
```

Sample Data:

EVID	PID
EV00001	PE0000008
EV00003	PE0000008
EV00004	PE0000008

CustomerLineItems Table

Functional Dependencies:

PRID, COID → Quantity

SQL CREATE statement:

```
CREATE TABLE CustomerLineItems (  
    COID      char (12)      not null,  
    PRID      char (8)       not null,  
    Quantity  int           not null,  
    primary key (COID, PRID),  
    foreign key (COID) references CustomerOrders (COID),  
    foreign key (PRID) references Products (PRID)  
)
```

Sample Data:

COID	PRID	Quantity
-----	-----	-----
COD000000001	PR000001	1
COD000000001	PR000004	1
COD000000002	PR000002	2
COD000000002	PR000008	1
COD000000003	PR000005	3
COD000000003	PR000007	1

Services Table

Functional Dependencies:

PRID → ServiceName, Description

SQL CREATE statement:

```
CREATE TABLE Services (  
    PRID          char (8)          not null,  
    ServiceName   varchar (20)      not null,  
    Description   varchar(30),  
primary key (PRID),  
foreign key (PRID) references Products (PRID)  
)
```

Sample Data:

PRID	ServiceName	Description
PR000011	Edge Sharpening	Sharpen Edges
PR000012	Base Refinish	P-Text and full referb of Base
PR000013	Base Wax	Wax Base

Snowboards Table

Functional Dependencies:

PRID → DeckColor, BaseColor, ProRider, OverallLengthMM, EffectiveEdgeMM, SideWall, WaistWidthMM, BindingPattern

SQL CREATE statement:

```
CREATE TABLE Snowboards (  
    PRID                char (8)                not null,  
    DeckColor           varchar (20)            not null,  
    BaseColor           varchar (20)            not null default ('white'),  
    ProRider            varchar (30),  
    OverallLengthMM     numeric (4,0)           not null,  
    EffectiveEdgeMM     numeric (4,0)           not null,  
    SideWall            varchar (20)            not null check (SideWall='Capped' or SideWall='Cut') default  
('Capped'),  
    WaistWidthMM        numeric (4,0)           not null,  
    BindingPattern       char (6) not null check (BindingPattern='3 Hole' or BindingPattern='4 Hole') default ('3  
Hole'),  
    primary key (PRID),  
    foreign key (PRID) references Products (PRID)  
)
```

Sample Data:

PRID	DeckColor	BaseColor	ProRider	OverallLengthMM	EffectiveEdgeMM	SideWall	WaistWidthMM	BindingPattern
PR000001	Black	White	Jenna Jameson	1580	1370	Capped	248	3 Hole
PR000002	Blue	White	Ross Powers	1560	1340	Capped	242	3 Hole
PR000007	Black	White	Chewbacca	1360	1170	Capped	242	3 Hole

SupplyLineItems Table

Functional Dependencies:

PRID, SOID → Quantity

SQL CREATE statement:

```
CREATE TABLE SupplyLineItems (  
    SOID      char (12)      not null,  
    PRID      char (8)       not null,  
    Quantity  int           not null,  
    primary key (SOID, PRID),  
    foreign key (SOID) references SupplyOrders (SOID),  
    foreign key (PRID) references Products (PRID)  
)
```

Sample Data:

SOID	PRID	Quantity
-----	-----	-----
SID000000001	PR000002	2
SID000000001	PR000004	6
SID000000001	PR000008	10
SID000000002	PR000004	6
SID000000003	PR000005	15
SID000000003	PR000009	15
SID000000004	PR000001	3

Reports:

Promotional Mailing Report

```
select distinct p.LastName as "Last Name", p.FirstName as "First Name", p.Address as "Street",
z.City as "City", z.State as "State", p.ZIP as "Zip-Code"
  from People p, Customers c, ZipCodes z, Employees e
  where p.PID = c.PID
        and p.ZIP = z.ZIP
  order by p.LastName
```

Last Name	First Name	Street	City	State	Zip-Code
Doctor	Love	876 NaughtyTime Blvd	Poughkeepsie	NY	12601
Dover	Ben	34 Pie Ln	Wappingers Falls	NY	12590
Matheus	Big "Hank"	127 Spackenkill Rd	Poughkeepsie	NY	12603
Matheus	Joshua	172 Wilbur Blvd	Poughkeepsie	NY	12603
Vader	Darth	327 DeathStar Drive	Poughkeepsie	NY	12603

Snowboard Inventory Report

```
select  p.Model as "Snowboard Model", v.CompanyName as "Company Name", s.OverallLengthMM as
"Length (mm)", p.RetailUSD as "Price (USD)", p.QuantityOnHand as "Amount in Stock"
  from Products p, Vendors v, Snowboards s
  where p.VID = v.VID
        and p.PRID = s.PRID
        and p.PType like 'SB'
  order by p.Model
```

Snowboard Model	Company Name	Length (mm)	Price (USD)	Amount in Stock
Floater	Ride	1360	420.0000	3

Jameson	Sims	1580	500.0000	3
Powers	Burton	1560	550.0000	2

Accessories Inventory Report

```

select  distinct p.Model as "Model", pt.PType as "Accessory", a.Color, v.CompanyName as
"Company Name", p.RetailUSD as "Price (USD)", p.QuantityOnHand as "Amount in Stock"
  from Products p, Vendors v, Accessories a, PTypes pt
  where p.VID = v.VID and p.PRID = a.PRID and p.PType = pt.PType
  order by p.Model

```

Model	Accessory	Color	Company Name	Price (USD)	Amount in Stock
AK Outland	PT	GR	Burton	200.0000	6
Big Foot	BT	OG	Volcom	180.0000	10
Big Hands	GL	RD	Burton	60.0000	10
Blinder	GG	BK	Jeenyus	80.0000	12
Ice Man	GG	GN	RobotFood	90.0000	15
Mr Freeze	JK	BL	Jeenyus	150.0000	5
Slick Willy	WX	PK	RobotFood	12.0000	15

Views:

Inventory View

```
Create View Accessory_Inventory (Model, Product, ColorCode, Company, Price, Amount) as
select distinct p.Model as "Model", pt.PType as "Accessory", a.Color, v.CompanyName as
"Company Name", p.RetailUSD as "Price (USD)", p.QuantityOnHand as "Amount in Stock"
  from Products p, Vendors v, Accessories a, PTypes pt
  where p.VID = v.VID and p.PRID = a.PRID and p.PType = pt.PType
```

```
Select *
From Accessory_Inventory
```

Model	Product	ColorCode	Company	Price	Amount
AK Outland	PT	GR	Burton	200.0000	6
Big Foot	BT	OG	Volcom	180.0000	10
Big Hands	GL	RD	Burton	60.0000	10
Blinder	GG	BK	Jeenyus	80.0000	12
Ice Man	GG	GN	RobotFood	90.0000	15
Mr Freeze	JK	BL	Jeenyus	150.0000	5
Slick Willy	WX	PK	RobotFood	12.0000	15

Employee View

```
Create View Employee_Records (LastName, FirstName, EmployeeID, SSN, Dept, SalaryUSD, HireDate,
PhoneNumber) as
select distinct p.LastName, p.FirstName, e.PID, e.SSN, d.DeptName, e.Salary, e.HireDate,
p.Phone
```

from People p, Employees e, DeptCodes d
where p.PID = e.PID and e.DeptID = d.DeptID

Select *
From Employee_Records
Order by LastName

LastName	FirstName	EmployeeID	SSN	Dept	SalaryUSD	HireDate	PhoneNumber
Bond	James	PE0000003	100770007	Purchasing	60000.0000	1998-10-07	5185559099
Jameson	Jenna	PE0000008	111111117	Pro	90000.0000	1996-06-06	8454716666
Matheus	Joshua	PE0000001	100628999	Sales	32000.0000	2000-11-27	8454719099
Matheus	Carolyn	PE0000002	118669073	Accounting	80000.0000	2000-11-27	8454719099
Matheus	Chewbacca	PE0000009	999897999	Executive	300000.0000	2003-01-01	8454719099
Matheus	Big "Hank"	PE0000010	152639877	Sales	50000.0000	1999-04-25	8454626311

Security:

The security in this database ensures that the regular user access (MagiK_User) is prohibited to viewing and adding data while at the same time restricting any operations at all on the **Employees** table where salary information is stored.

The administrator of the database has full access to all tables and can read, write, modify, or delete data.
The general user role does not have delete access to any of the tables and all functions are revoked on the employee table.

Administrator role:

Revoke all on People from MagiK_Admin;
Revoke all on Employees from MagiK_Admin;
Revoke all on Customers from MagiK_Admin;
Revoke all on ProRiders from MagiK_Admin;
Revoke all on Products from MagiK_Admin;
Revoke all on Snowboards from MagiK_Admin;
Revoke all on Accessories from MagiK_Admin;
Revoke all on Services from MagiK_Admin;
Revoke all on PTypes from MagiK_Admin;
Revoke all on DeptCodes from MagiK_Admin;
Revoke all on ZIPCodes from MagiK_Admin;
Revoke all on Events from MagiK_Admin;
Revoke all on Vendors from MagiK_Admin;
Revoke all on Competes from MagiK_Admin;
Revoke all on CustomerOrders from MagiK_Admin;
Revoke all on CustomerLineItems from MagiK_Admin;
Revoke all on SupplyOrders from MagiK_Admin;
Revoke all on SupplyLineItems from MagiK_Admin;

Grant insert, update, delete, select on People to MagiK_Admin;
Grant insert, update, delete, select on Employees to MagiK_Admin;
Grant insert, update, delete, select on Customers to MagiK_Admin;

**Grant insert, update, delete, select on ProRiders to MagiK_Admin;
Grant insert, update, delete, select on Products to MagiK_Admin;
Grant insert, update, delete, select on Snowboards to MagiK_Admin;
Grant insert, update, delete, select on Accessories to MagiK_Admin;
Grant insert, update, delete, select on Services to MagiK_Admin;
Grant insert, update, delete, select on PTypes to MagiK_Admin;
Grant insert, update, delete, select on DeptCodes to MagiK_Admin;
Grant insert, update, delete, select on Events to MagiK_Admin;
Grant insert, update, delete, select on ZIPCodes to MagiK_Admin;
Grant insert, update, delete, select on Vendors to MagiK_Admin;
Grant insert, update, delete, select on Competes to MagiK_Admin;
Grant insert, update, delete, select on CustomerOrders to MagiK_Admin;
Grant insert, update, delete, select on CustomerLineItems to MagiK_Admin;
Grant insert, update, delete, select on SupplyOrders to MagiK_Admin;
Grant insert, update, delete, select on SupplyLineItems to MagiK_Admin;**

User Role:

**Revoke all on People from MagiK_User;
Revoke all on Employees from MagiK_User;
Revoke all on Customers from MagiK_User;
Revoke all on ProRiders from MagiK_User;
Revoke all on Products from MagiK_User;
Revoke all on Snowboards from MagiK_User;
Revoke all on Accessories from MagiK_User;
Revoke all on Services from MagiK_User;
Revoke all on PTypes from MagiK_User;
Revoke all on DeptCodes from MagiK_User;
Revoke all on ZIPCodes from MagiK_User;
Revoke all on Events from MagiK_User;
Revoke all on Vendors from MagiK_User;
Revoke all on Competes from MagiK_User;
Revoke all on CustomerOrders from MagiK_User;
Revoke all on CustomerLineItems from MagiK_User;**

**Revoke all on SupplyOrders from MagiK_User;
Revoke all on SupplyLineItems from MagiK_User;**

**Grant insert, update, select on People to MagiK_User;
Grant insert, update, select on Customers to MagiK_User;
Grant insert, update, select on ProRiders to MagiK_User;
Grant insert, update, select on Products to MagiK_User;
Grant insert, update, select on Snowboards to MagiK_User;
Grant insert, update, select on Accessories to MagiK_User;
Grant insert, update, select on Services to MagiK_User;
Grant insert, update, select on PTypes to MagiK_User;
Grant insert, update, select on DeptCodes to MagiK_User;
Grant insert, update, select on Events to MagiK_User;
Grant insert, update, select on ZIPCodes to MagiK_User;
Grant insert, update, select on Vendors to MagiK_User;
Grant insert, update, select on Competes to MagiK_User;
Grant insert, update, select on CustomerOrders to MagiK_User;
Grant insert, update, select on CustomerLineItems to MagiK_User;
Grant insert, update, select on SupplyOrders to MagiK_User;
Grant insert, update, select on SupplyLineItems to MagiK_User;**

Looking Forward:

Some additions that would be useful before this database reaches full production for Magik Snowboards are as follows:

- Use of a Stored Procedure to generate unique primary keys
- More granular security and possibly more user groups than “user” and “administrator”
- More Views to provide high level security and assist in daily data-mining in the store
- More Reports to help management get a handle on specific aspects of the entire operation