Introduction to Database Design

Database Design Agenda

- **Introductions**
- **General Design Considerations**
- **Entity-Relationship Model**
- **Normalization**
- **Overview of SQL**
- **Star Schemas**
- **Additional Information**
- **Q&A**

General Design Considerations

- **Users**
- **Application Requirements**
- **Legacy Systems/Data**

Users

- **Who are they?**
  - **Administrative**
  - **Scientific**
  - **Technical**
- **Impact**
  - **Access Controls**
  - **Interfaces**
  - **Service levels**

Application Requirements

- **What kind of database?**
  - **OnLine Analytical Processing (OLAP)**
  - **OnLine Transactional Processing (OLTP)**
- **Budget**
- **Platform / Vendor**
- **Workflow?**
  - **order of operations**
  - **error handling**
  - **reporting**

Legacy Systems/Data

- **What systems are currently in place?**
- **Where does the data come from?**
- **How is it generated?**
- **What format is it in?**
- **What is the data used for?**
- **Which parts of the system must remain static?**

Entity - Relationship Model

A logical design method which emphasizes simplicity and readability.

- **Basic objects of the model are:**
  - **Entities**
  - **Relationships**
  - **Attributes**

Entities

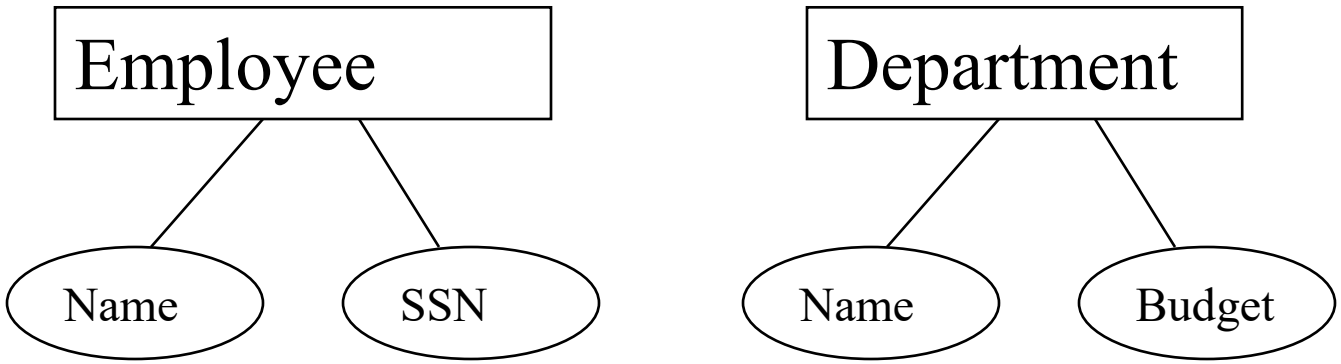
Data objects detailed by the information in the database.

- **Denoted by rectangles in the model.**

Attributes

Characteristics of entities or relationships.

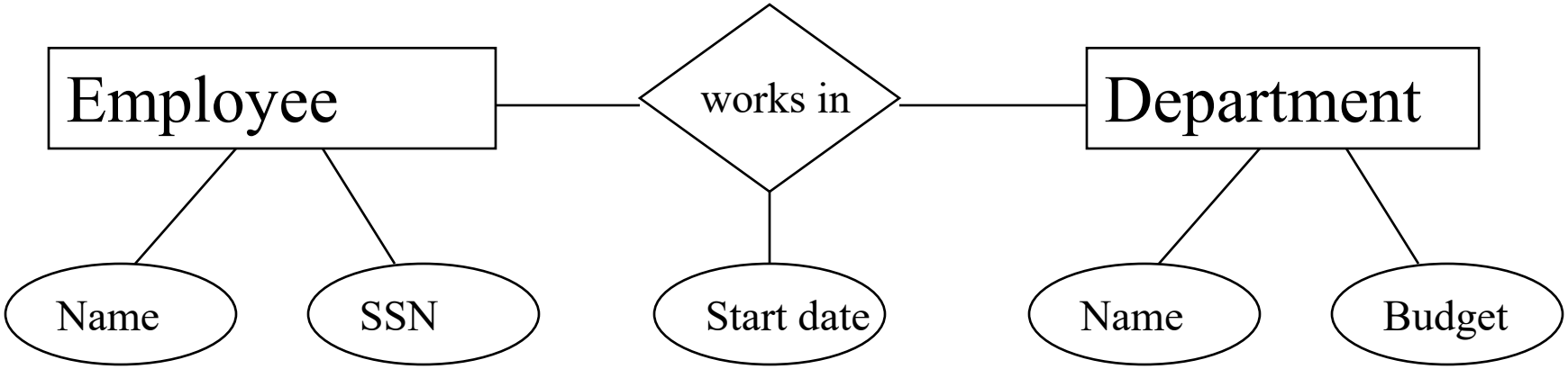
- **Denoted by ellipses in the model.**



Relationships

Represent associations between entities.

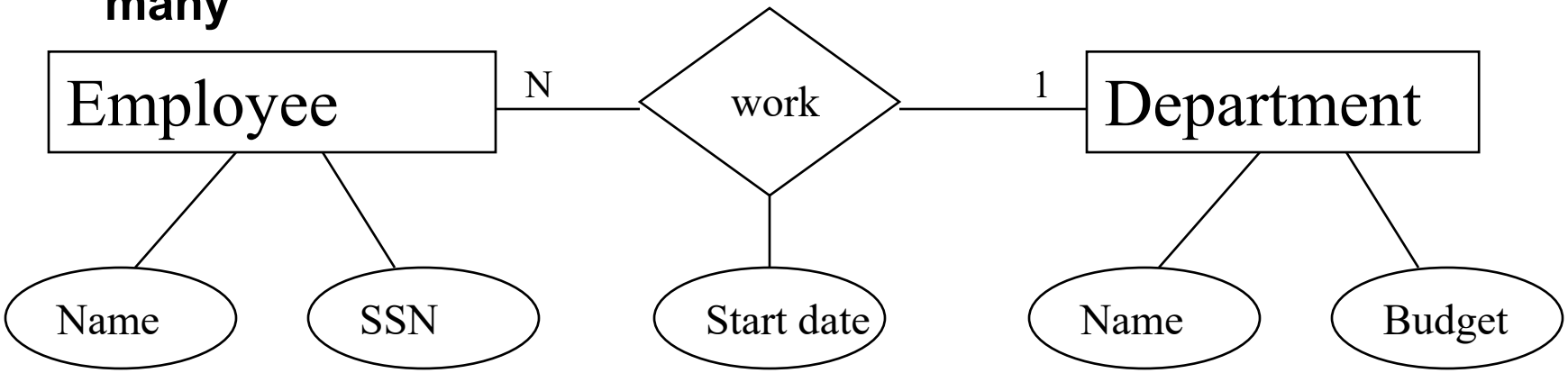
- **Denoted by diamonds in the model.**



Relationship Connectivity

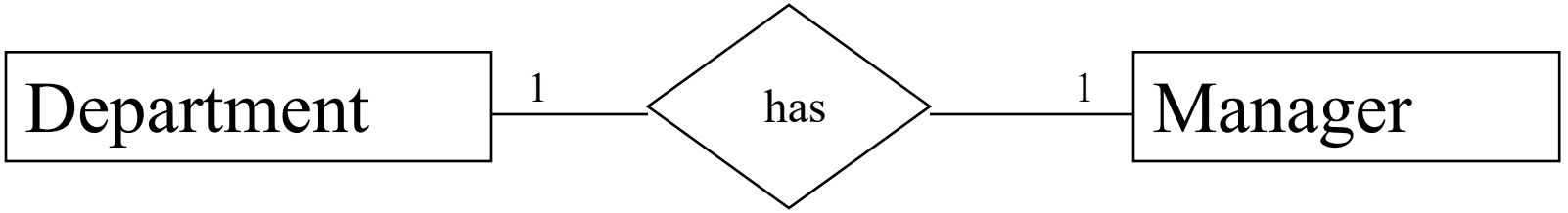
Constraints on the mapping of the associated entities in the relationship.

- **Denoted by variables between the related entities.**
- **Generally, values for connectivity are expressed as "one" or "many"**

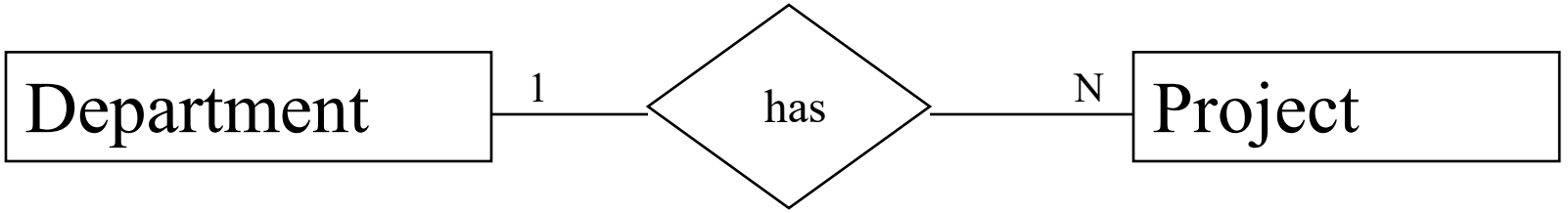


Connectivity

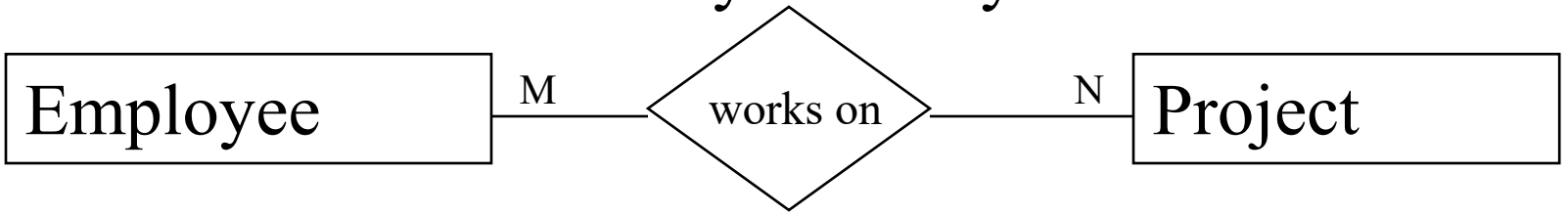
one-to-one



one-to-many



many-to-many



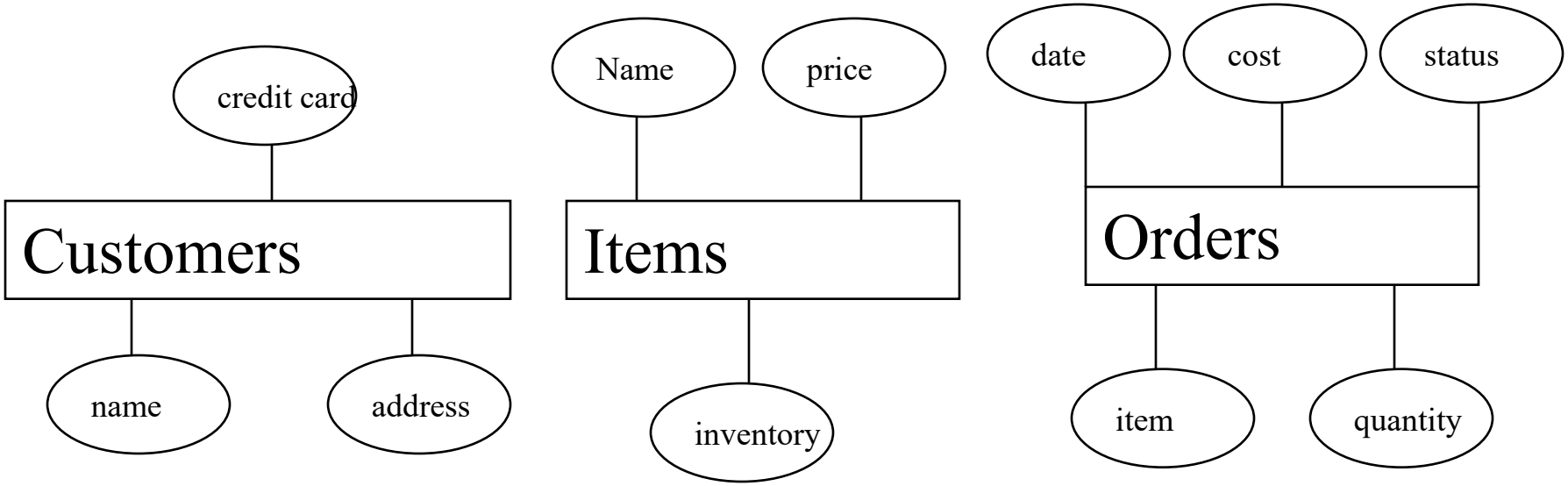
ER example

Retailer wants to create an online webstore.

- **The retailer requires information on:**
  - **Customers**
  - **Items**
  - **Orders**

Webstore Entities & Attributes

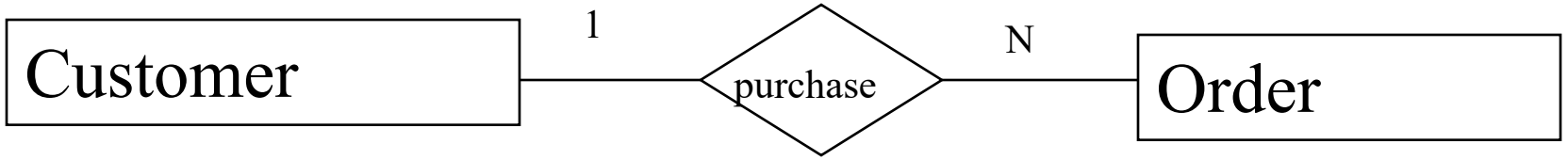
- **Customers - name, credit card, address**
- **Items - name, price, inventory**
- **Orders - item, quantity, cost, date, status**



Webstore Relationships

Identify the relationships.

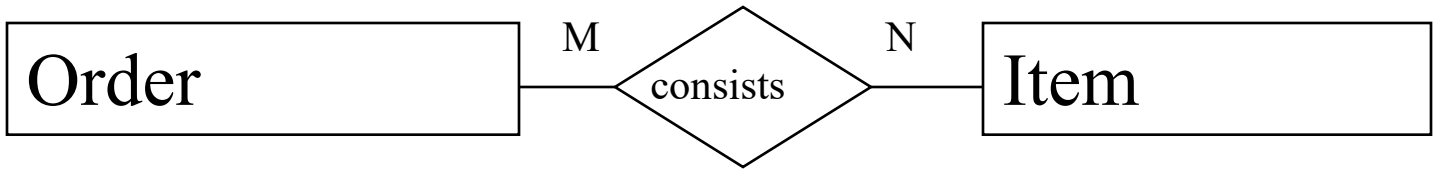
- **The orders are recorded each time a customer purchases items, so the customer and order entities are related.**
  - **Each customer may make several purchases so the relationship is one-to-many**



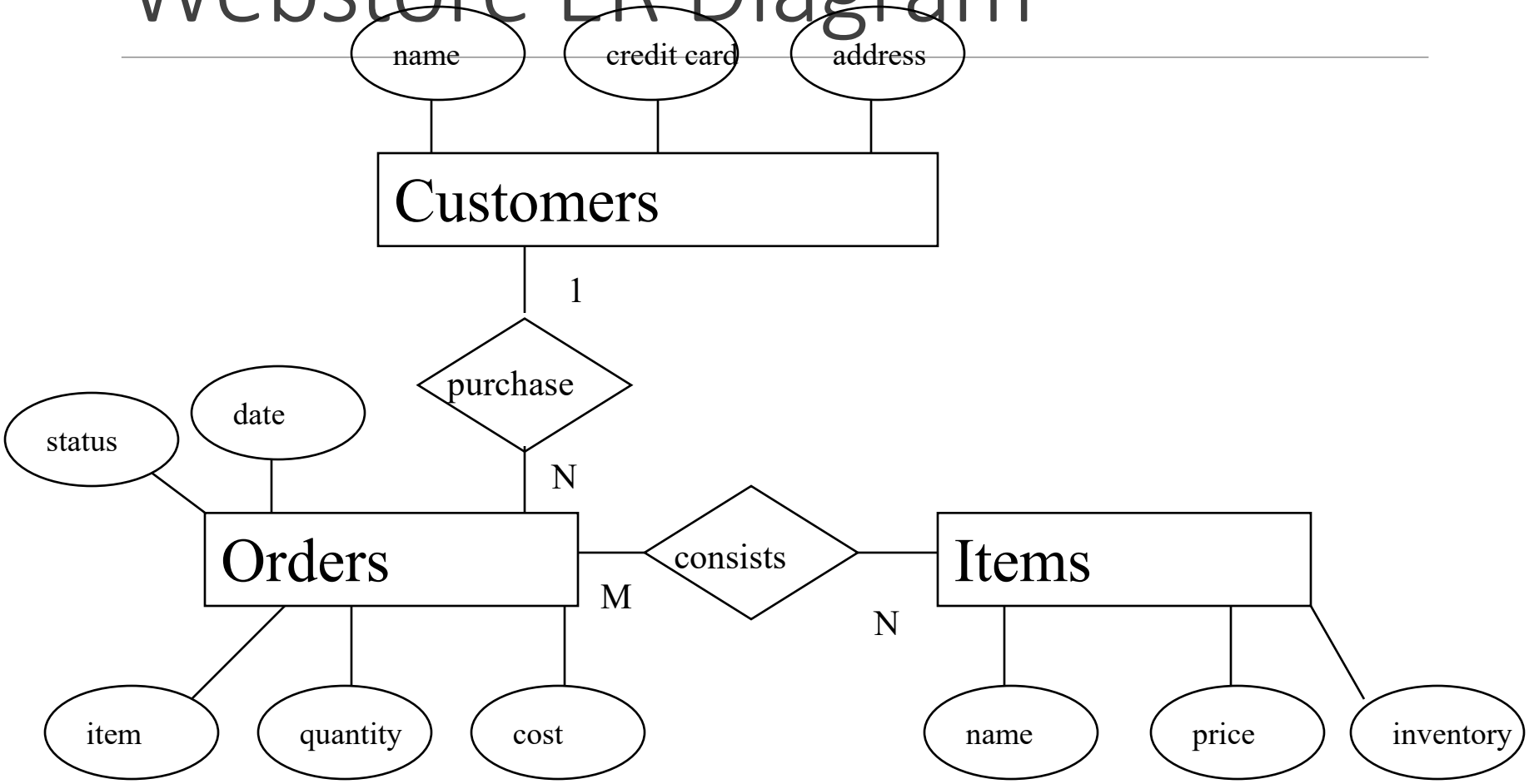
Webstore Relationships

Identify the relationships.

- **The order consists of the items a customer purchases but each item can be found in multiple orders.**
  - **Since a customer can purchase multiple items and make multiple orders the relationship is many to many.**



Webstore ER Diagram



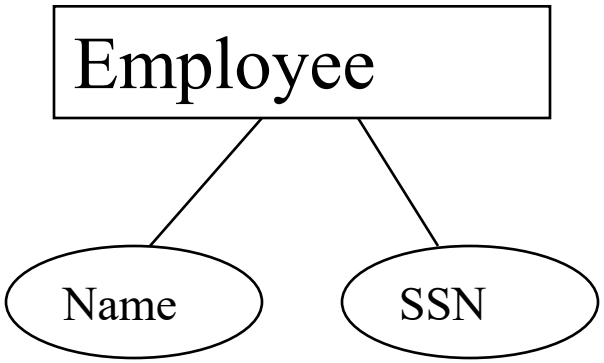
Logical Design to Physical Design

Creating relational SQL schemas from entity-relationship models.

- **Transform each entity into a table with the key and its attributes.**
- **Transform each relationship as either a relationship table (many-to-many) or a "foreign key" (one-to-many and many-to-many).**

Entity tables

Transform each entity into a table with a key and its attributes.

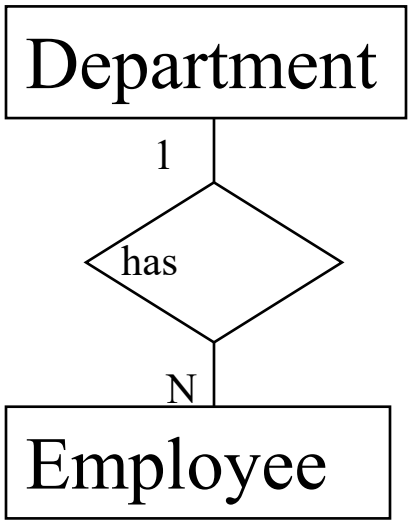



```
create table employee
        (emp_no number,
        name varchar2(256),
        ssn number,
        primary key (emp_no));
```

Foreign Keys

Transform each one-to-one or one-to-many relationship as a "foreign key".

- **Foreign key is a reference in the child (many) table to the primary key of the parent (one) table.**




```
create table department
      (dept_no number,
      name varchar2(50),
      primary key (dept_no));
  create table employee
      (emp_no number,
      dept_no number,
      name varchar2(256),
      ssn number,
      primary key (emp_no),
      foreign key (dept_no) references department);
```

Foreign Key

Department

dept_no	Name
1	Accounting
2	Human Resources
3	IT

Employee

emp_no	dept_no	Name
1	2	Nora Edwards
2	3	Ajay Patel
3	2	Ben Smith
4	1	Brian Burnett
5	3	John O'Leary
6	3	Julia Lenin

Accounting has 1 employee:

Brian Burnett

Human Resources has 2 employees:

Nora Edwards
Ben Smith

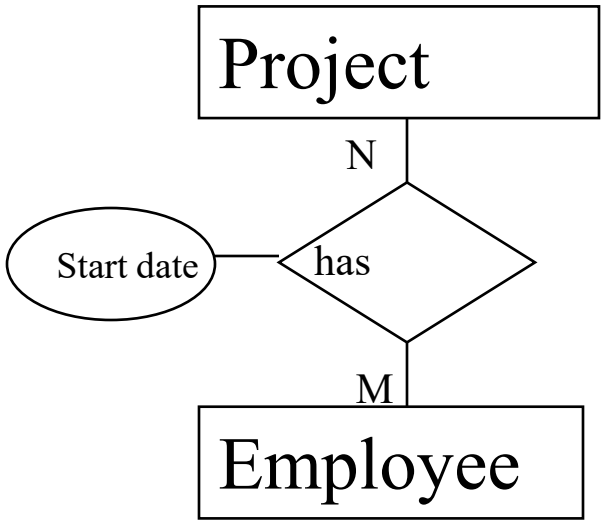
IT has 3 employees:

Ajay Patel
John O'Leary
Julia Lenin

Many-to-Many tables

Transform each many-to-many relationship as a table.

- **The relationship table will contain the foreign keys to the related entities as well as any relationship attributes.**




```
create table project_employee_details
        (proj_no number,
        emp_no number,
        start_date date,
        primary key (proj_no, emp_no),
        foreign key (proj_no) references project
        foreign key (emp_no) references employee);
```

Many-to-Many tables

Project

proj_no	Name
1	Employee Audit
2	Budget
3	Intranet

Project_employee_details

proj_no	emp_no	start_date
1	4	4/7/03
3	6	8/12/02
3	5	3/4/01
2	6	11/11/02
3	2	12/2/03
2	1	7/21/04

Employee

emp_no	dept_no	Name
1	2	Nora Edwards
2	3	Ajay Patel
3	2	Ben Smith
4	1	Brian Burnett
5	3	John O'Leary
6	3	Julia Lenin

Employee Audit has 1 employee:

Brian Burnett

Budget has 2 employees:

Julia Lenin
Nora Edwards

Intranet has 3 employees:

Julia Lenin
John O'Leary
Ajay Patel

Normalization

A logical design method which minimizes data redundancy and reduces design flaws.

- **Consists of applying various “normal” forms to the database design.**
- **The normal forms break down large tables into smaller subsets.**

First Normal Form (1NF)

Each attribute must be atomic

- No repeating columns within a row.
- No multi-valued columns.

1NF simplifies attributes

- Queries become easier.

1NF

Employee (unnormalized)

emp_no	name	dept_no	dept_name	skills
1	Kevin Jacobs	201	R&D	C, Perl, Java
2	Barbara Jones	224	IT	Linux, Mac
3	Jake Rivera	201	R&D	DB2, Oracle, Java

Employee (1NF)

emp_no	name	dept_no	dept_name	skills
1	Kevin Jacobs	201	R&D	C
1	Kevin Jacobs	201	R&D	Perl
1	Kevin Jacobs	201	R&D	Java
2	Barbara Jones	224	IT	Linux
2	Barbara Jones	224	IT	Mac
3	Jake Rivera	201	R&D	DB2
3	Jake Rivera	201	R&D	Oracle
3	Jake Rivera	201	R&D	Java

Second Normal Form (2NF)

Each attribute must be functionally dependent on the primary key.

- Functional dependence - the property of one or more attributes that uniquely determines the value of other attributes.
- Any non-dependent attributes are moved into a smaller (subset) table.

2NF improves data integrity.

- Prevents update, insert, and delete anomalies.

Functional Dependence

Employee (1NF)

emp_no	name	dept_no	dept_name	skills
1	Kevin Jacobs	201	R&D	C
1	Kevin Jacobs	201	R&D	Perl
1	Kevin Jacobs	201	R&D	Java
2	Barbara Jones	224	IT	Linux
2	Barbara Jones	224	IT	Mac
3	Jake Rivera	201	R&D	DB2
3	Jake Rivera	201	R&D	Oracle
3	Jake Rivera	201	R&D	Java

Name, dept_no, and dept_name are functionally dependent on emp_no. **(emp_no -> name, dept_no, dept_name)**

Skills is not functionally dependent on emp_no since it is not unique to each emp_no.

2NF

Employee (1NF)

emp_no	name	dept_no	dept_name	skills
1	Kevin Jacobs	201	R&D	C
1	Kevin Jacobs	201	R&D	Perl
1	Kevin Jacobs	201	R&D	Java
2	Barbara Jones	224	IT	Linux
2	Barbara Jones	224	IT	Mac
3	Jake Rivera	201	R&D	DB2
3	Jake Rivera	201	R&D	Oracle
3	Jake Rivera	201	R&D	Java

Employee (2NF)

emp_no	name	dept_no	dept_name
1	Kevin Jacobs	201	R&D
2	Barbara Jones	224	IT
3	Jake Rivera	201	R&D

Skills (2NF)

emp_no	skills
1	C
1	Perl
1	Java
2	Linux
2	Mac
3	DB2
3	Oracle
3	Java

Data Integrity

Employee (1NF)

emp_no	name	dept_no	dept_name	skills
1	Kevin Jacobs	201	R&D	C
1	Kevin Jacobs	201	R&D	Perl
1	Kevin Jacobs	201	R&D	Java
2	Barbara Jones	224	IT	Linux
2	Barbara Jones	224	IT	Mac
3	Jake Rivera	201	R&D	DB2
3	Jake Rivera	201	R&D	Oracle
3	Jake Rivera	201	R&D	Java

- Insert Anomaly - adding null values. eg, inserting a new department does not require the primary key of emp_no to be added.
- Update Anomaly - multiple updates for a single name change, causes performance degradation. eg, changing IT dept_name to IS
- Delete Anomaly - deleting wanted information. eg, deleting the IT department removes employee Barbara Jones from the database

Third Normal Form (3NF)

Remove transitive dependencies.

- Transitive dependence - two separate entities exist within one table.
- Any transitive dependencies are moved into a smaller (subset) table.

3NF further improves data integrity.

- Prevents update, insert, and delete anomalies.

Transitive Dependence

Employee (2NF)

emp_no	name	dept_no	dept_name
1	Kevin Jacobs	201	R&D
2	Barbara Jones	224	IT
3	Jake Rivera	201	R&D

Dept_no and dept_name are functionally dependent on emp_no however, department can be considered a separate entity.

3NF

Employee (2NF)

emp_no	name	dept_no	dept_name
1	Kevin Jacobs	201	R&D
2	Barbara Jones	224	IT
3	Jake Rivera	201	R&D

Employee (3NF)

emp_no	name	dept_no
1	Kevin Jacobs	201
2	Barbara Jones	224
3	Jake Rivera	201

Department (3NF)

dept_no	dept_name
201	R&D
224	IT

Other Normal Forms

Boyce-Codd Normal Form (BCNF)

- Strengthens 3NF by requiring the keys in the functional dependencies to be superkeys (a column or columns that uniquely identify a row)

Fourth Normal Form (4NF)

- Eliminate trivial multivalued dependencies.

Fifth Normal Form (5NF)

- Eliminate dependencies not determined by keys.

Normalizing our webstore (1NF)

orders

order_id	cust_id	item_id	quantity	cost	date	status
405	45	34	2	100	2/306	shipped
405	45	35	1	50	2/306	shipped
405	45	56	3	75	2/306	shipped
408	78	56	2	50	3/5/06	refunded
410	102	72	2	150	3/10/06	shipped
410	102	81	1	175	3/10/06	shipped

items

item_id	name	price	inventory
34	sweater red	50	21
35	sweater blue	50	10
56	t-shirt	25	76
72	jeans	75	5
81	jacket	175	9

customers

cust_id	name	address	credit_card_num	credit_card_type
45	Mike Speedy	123 A St.	45154	visa
45	Mike Speedy	123 A St.	32499	mastercard
45	Mike Speedy	123 A St.	12834	discover
78	Frank Newmon	2 Main St.	45698	visa
102	Joe Powers	343 Blue Blvd.	94065	mastercard
102	Joe Powers	343 Blue Blvd.	10532	discover

Normalizing our webstore (2NF & 3NF)

customers

cust_id	name	address
45	Mike Speedy	123 A St.
78	Frank Newmon	2 Main St.
102	Joe Powers	343 Blue Blvd.

credit_cards

cust_id	num	type
45	45154	visa
45	32499	mastercard
45	12834	discover
78	45698	visa
102	94065	mastercard
102	10532	discover

Normalizing our webstore (2NF & 3NF)

items

item_id	name	price	inventory
34	sweater red	50	21
35	sweater blue	50	10
56	t-shirt	25	76
72	jeans	75	5
81	jacket	175	9

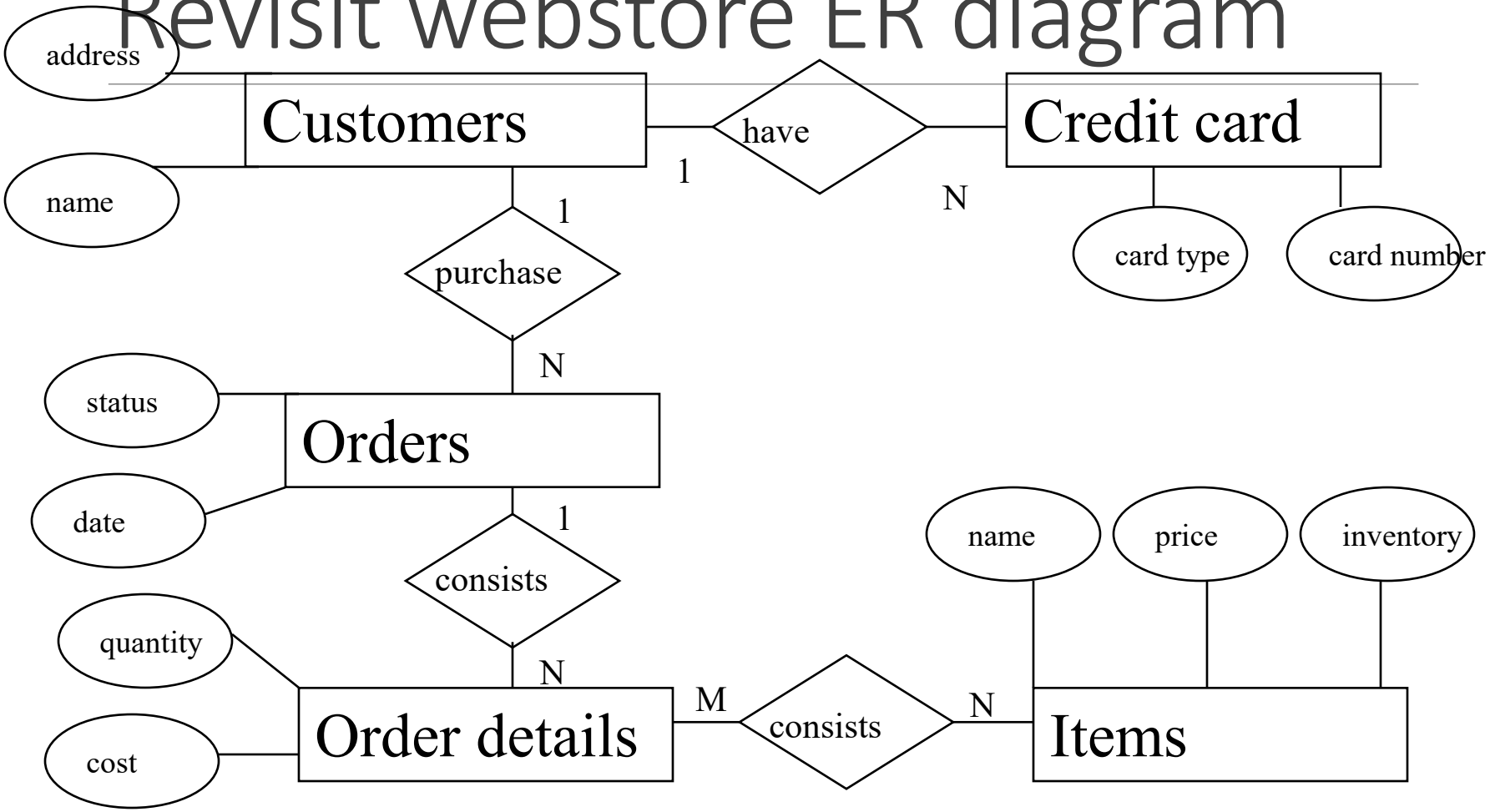
orders

order_id	cust_id	date	status
405	45	2/306	shipped
408	78	3/5/06	refunded
410	102	3/10/06	shipped

order details

order_id	item_id	quantity	cost
405	34	2	100
405	35	1	50
405	56	3	75
408	56	2	50
410	72	2	150
410	81	1	175

Revisit webstore ER diagram



Structured Query Language

SQL is the standard language for data definition and data manipulation for relational database systems.

- **Nonprocedural**
- **Universal**

Data Definition Language

The aspect of SQL that defines and manipulates objects in a database.

- **create tables**
- **alter tables**
- **drop tables**
- **create views**