Van Whinkle Database Design

Caitlin Ross

ross0272@algonquinlive.com

Student Number 040750891

Assignment #1

Course Code CST2355 Section 451

Table of Contents

Table Data (MySQL)

ADDRESS Table

AddressID	AddressLine1	AddressLine2	City	Province	PostalCode
1	NULL	NULL	Windsor	Ontario	NULL
2	NULL	NULL	Toronto	Ontario	NULL
3	NULL	NULL	London	Ontario	NULL
4	NULL	NULL	Vancouver	British Columbia	NULL
5	NULL	NULL	Winnipeg	Manitoba	NULL
6	NULL	NULL	Woodstock	Ontario	NULL
NULL	NULL	NULL	NULL	NULL	NULL

CHARGE Table

ReservationID	ChargeType	Value	Quantity
1	Deposit: Smoker	50.00	1
1	Room Rate: Standard	50.00	14
1	Damage: Smoke	25.00	1
1	Damage: Carpet Burns	30.00	1
2	Room Rate: Student	30.00	1
2	Deposit: Non-Smoker Loyalty	0.00	1
3	Room Rate: Loyalty	35.00	1
3	Deposit: Smoker Loyalty	20.00	1
4	Room Rate: Loyalty	35.00	1
4	Deposit: Smoker Loyalty	20.00	1
5	Room Rate: Loyalty	35.00	1
5	Deposit: Smoker Loyalty	20.00	1
6	Room Rate: Loyalty	35.00	1
6	Deposit: Smoker Loyalty	20.00	1
7	Deposit: Manager	0.00	1
7	Room Rate: Manager	0.00	1
8	Deposit: Manager	0.00	1
8	Room Rate: Manager	0.00	1
9	Deposit: Manager	0.00	1
9	Room Rate: Manager	0.00	1
10	Deposit: Manager	0.00	1
10	Room Rate: Manager	0.00	1
1	Deposit Return	50.00	1
3	Deposit Return	20.00	1
4	Deposit Return	20.00	1
5	Deposit Return	20.00	1
6	Deposit Return	20.00	1

CUSTOMER Table

CustomerID	FirstName	LastName	Notes
1	Kristoff	Kurn	Smoker deposit($50) and full rate($50/day)
2	Billy	Elliot	Loyalty deposit($0) and student rate($30/day)
3	Justin	Hackman	Loyalty smoker deposit($20) and loyalty rate($...
4	Deena	Donor	Charge to Van Whinkle
NULL	NULL	NULL	NULL

CUSTOMERADDRESS Table

CustomerID	AddressID	IsCurrent
1	4	1
2	5	1
2	2	0
3	6	1

EMPLOYEE Table

EmployeeID	FirstName	LastName	HireDate	Position	ManagerID	WorkLocation	PayFrequency	PayRate
1	Bob	Smith	2000-01-01	Manager	4	London	Yearly	80000
2	Bob	Smith	2015-01-01	Desk Clerk	1	London	Hourly	15
3	Tanya	Duncan	2010-01-01	Maintenance	1	London	Monthly	5000
4	Deena	Donor	NULL	Operations Manager	NULL	NULL	Hourly	0

EMPLOYEEADDRESS Table

EmployeeID	AddressID	IsCurrent
1	1	1
2	2	1
3	3	1

EMPLOYEECUSTOMER Table

EmployeeID	CustomerID
4	4

RESERVATION Table

CustomerID	ReservationDate	CheckInDate	CheckOutDate	Location	ReservationID
1	2020-01-03	2020-02-03	2020-02-17	London	1
2	2020-01-20	2020-01-20	2020-01-21	London	2
3	NULL	2020-02-02	2020-02-03	London	3
3	NULL	2020-02-08	2020-02-09	London	4
3	NULL	2020-02-17	2020-02-18	London	5
3	NULL	2020-02-28	2020-03-01	London	6
4	NULL	2020-01-01	2020-01-02	London	7
4	NULL	2020-02-01	2020-02-02	London	8
4	NULL	2020-03-01	2020-03-02	London	9
4	NULL	2020-04-01	2020-04-02	London	10

SQL Queries

Any blank fields in CUSTOMER table, sorted Z-A on last name

```
SELECT *
FROM CUSTOMER
WHERE
    CustomerID IS NULL
    OR FirstName IS NULL
    OR LastName IS NULL
    OR Notes IS NULL
ORDER BY LastName DESC;
```

Differences in SQL Syntax

- MySQL
  - No change
- SQL Server
  - No change
- Oracle
  - Replace CUSTOMER with ASSIGN_1.CUSTOMER on line 2

Any blank fields in CUSTOMER table, sorted A-Z on first name

```
SELECT *
FROM CUSTOMER
WHERE CustomerID IS NULL
    OR FirstName IS NULL
    OR LastName IS NULL
    OR Notes IS NULL
ORDER BY FirstName ASC;
```

Differences in SQL Syntax

- MySQL
  - No change
- SQL Server
  - No change
- Oracle
  - Replace CUSTOMER with ASSIGN_1.CUSTOMER on line 2

Show customers who have had 2+ day stays

```
SELECT
    CUSTOMER.FirstName,
    CUSTOMER.LastName,
    Stay.Quantity AS LengthOfStay
FROM CUSTOMER
    JOIN
        (SELECT
            CHARGE.Quantity,
            RESERVATION.CustomerID
        FROM RESERVATION
            JOIN CHARGE
            ON RESERVATION.ReservationID=CHARGE.ReservationID
        ) AS Stay
    ON CUSTOMER.CustomerID = Stay.CustomerID
WHERE Stay.Quantity>=2;
```

Differences in SQL Syntax

- MySQL
  - No change
- SQL Server
  - No change
- Oracle
  - Replace CUSTOMER with ASSIGN_1.CUSTOMER on line 5
  - Replace RESERVATION with ASSIGN_1.RESERVATION on line 10
  - Replace CHARGE with ASSIGN_1.CHARGE on line 11
  - Remove AS keyword on line 13

Show customers who have had 4-day stays

```
SELECT
    CUSTOMER.FirstName,
    CUSTOMER.LastName,
    Stay.Quantity AS LengthOfStay
FROM CUSTOMER
    JOIN
        (SELECT
            CHARGE.Quantity,
            RESERVATION.CustomerID
        FROM RESERVATION
            JOIN CHARGE
            ON RESERVATION.ReservationID=CHARGE.ReservationID
        ) AS Stay
    ON CUSTOMER.CustomerID = Stay.CustomerID
WHERE Stay.Quantity=4;
```

Differences in SQL Syntax

- MySQL
    - No change
- SQL Server
    - No change
- Oracle
    - Replace CUSTOMER with ASSIGN_1.CUSTOMER on line 5
    - Replace RESERVATION with ASSIGN_1.RESERVATION on line 10
    - Replace CHARGE with ASSIGN_1.CHARGE on line 11
    - Remove AS keyword on line 13

Show customers who have had 6+ day stays

```
SELECT
    CUSTOMER.FirstName,
    CUSTOMER.LastName,
    Stay.Quantity AS LengthOfStay
FROM CUSTOMER
    JOIN
        (SELECT
            CHARGE.Quantity,
            RESERVATION.CustomerID
        FROM RESERVATION
            JOIN CHARGE
            ON RESERVATION.ReservationID=CHARGE.ReservationID
        ) AS Stay
    ON CUSTOMER.CustomerID = Stay.CustomerID
WHERE Stay.Quantity >=6;
```

Differences in SQL Syntax

- MySQL
  - No change
- SQL Server
  - No change
- Oracle
  - Replace CUSTOMER with ASSIGN_1.CUSTOMER on line 5
  - Replace RESERVATION with ASSIGN_1.RESERVATION on line 10
  - Replace CHARGE with ASSIGN_1.CHARGE on line 11
  - Remove AS keyword on line 13

Find stays costing over $50

```
SELECT
    CUSTOMER.FirstName,
    CUSTOMER.LastName,
     RESCOST.ChargeType,
     RESCOST.Fee,
     RESCOST.TotalCost
FROM CUSTOMER
    JOIN
        (SELECT
            RESERVATION.CustomerID,
            STAY.ChargeType,
            STAY.Fee,
            STAY.TotalCost
        FROM RESERVATION
            JOIN
                (SELECT
                    CHARGE.ReservationID,
                    CHARGE.ChargeType,
                    CHARGE.Value*CHARGE.Quantity AS Fee,
                    STAYTOTAL.TotalCost
                FROM CHARGE
                    JOIN
                        (SELECT
                            ReservationID,
                            SUM(Value*Quantity) AS TotalCost
                        FROM CHARGE
                        GROUP BY ReservationID
                        ) AS STAYTOTAL
                    ON CHARGE.ReservationID=STAYTOTAL.ReservationID
                ) AS STAY
            ON RESERVATION.ReservationID=STAY.ReservationID
        ) AS RESCOST
    ON CUSTOMER.CustomerID=RESCOST.CustomerID
WHERE RESCOST.TotalCost>=50.00
ORDER BY RESCOST.Fee DESC;
```

Differences in SQL Syntax

- MySQL
  - No change
- SQL Server
  - No change
- Oracle
  - Replace CUSTOMER with ASSIGN_1.CUSTOMER on line 7
  - Replace RESERVATION with ASSIGN_1.RESERVATION on line 14
  - Replace CHARGE with ASSIGN_1.CHARGE on line 21 and line 26
  - Remove AS keyword from lines 28, 30 and 32

Find stays where extra fees cost over $50

```
SELECT
    CUSTOMER.FirstName,
    CUSTOMER.LastName,
    RESCOST.ChargeType,
    RESCOST.Fee,
    RESCOST.TotalCost
FROM CUSTOMER
    JOIN
        (SELECT
            RESERVATION.CustomerID,
            STAY.ChargeType,
            STAY.Fee,
            STAY.TotalCost
        FROM RESERVATION
            JOIN
                (SELECT
                    CHARGE.ReservationID,
                    CHARGE.ChargeType,
                    CHARGE.Value*CHARGE.Quantity AS Fee,
                    STAYTOTAL.TotalCost
                FROM CHARGE
                    JOIN
                        (SELECT
                            ReservationID,
                            SUM(Value*Quantity) AS TotalCost
                        FROM CHARGE
                        WHERE CHARGE.ChargeType NOT LIKE '%Room Rate%'
                        GROUP BY ReservationID
                        ) AS STAYTOTAL
                    ON CHARGE.ReservationID=STAYTOTAL.ReservationID
                ) AS STAY
            ON RESERVATION.ReservationID=STAY.ReservationID
        ) AS RESCOST
    ON CUSTOMER.CustomerID=RESCOST.CustomerID
WHERE
    RESCOST.TotalCost>=50.00
    AND RESCOST.ChargeType NOT LIKE '%Room Rate%'
ORDER BY RESCOST.Fee DESC;
```

Differences in SQL Syntax

- MySQL
  - No change
- SQL Server
  - No change
- Oracle
  - Replace CUSTOMER with ASSIGN_1.CUSTOMER on line 7
  - Replace RESERVATION with ASSIGN_1.RESERVATION on line 14
  - Replace CHARGE with ASSIGN_1.CHARGE on line 21 and line 26
  - Remove AS keyword from lines 29, 31 and 33

Display employee table with manager's info

```
SELECT
    EMPMAN.FirstName,
    EMPMAN.LastName,
    EMPADDRESS.City,
    EMPMAN.ManagersFirstName,
    EMPMAN.ManagersLastName,
    MANADDRESS.City
FROM
    (SELECT
        emp.FirstName,
        emp.LastName,
        man.FirstName AS ManagersFirstName,
        man.LastName AS ManagersLastName,
        emp.EmployeeID AS empID,
        man.EmployeeID AS manID
    FROM EMPLOYEE emp
        LEFT JOIN EMPLOYEE man
        ON emp.ManagerID=man.EmployeeID
    ) AS EMPMAN
    LEFT JOIN
        (SELECT
            ADDRESS.City,
            EMPLOYEEADDRESS.EmployeeID
        FROM EMPLOYEEADDRESS
            LEFT JOIN ADDRESS
            ON ADDRESS.AddressID=EMPLOYEEADDRESS.AddressID
        ) AS EMPADDRESS
    ON EMPMAN.empID=EMPADDRESS.EmployeeID
    LEFT JOIN
        (SELECT
            ADDRESS.City,
            EMPLOYEEADDRESS.EmployeeID
        FROM EMPLOYEEADDRESS
            LEFT JOIN ADDRESS
            ON ADDRESS.AddressID=EMPLOYEEADDRESS.AddressID
        ) AS MANADDRESS
    ON EMPMAN.manID=MANADDRESS.EMPLOYEEID;
```

Differences in SQL Syntax

- MySQL
  - No change
- SQL Server
  - No change
- Oracle
  - Replace EMPLOYEE with ASSIGN_1.EMPLOYEE on lines 16 and 17
  - Replace EMPLOYEEADDRESS with ASSIGN_1.EMPLOYEEADDRESS on lines 24 and 33
  - Replace ADDRESS with ASSIGN_1.ADDRESS on lines 25 and 34
  - Remove AS keyword from lines 19, 27 and 36

Show all employees and customers from London and Winnipeg

```
SELECT
    CUSTOMER.FirstName AS FirstName,
    CUSTOMER.LastName AS LastName,
    C_ADDRESS.City AS City
FROM
    (SELECT
        CUSTOMERADDRESS.CustomerID,
        ADDRESS.City,
        CUSTOMERADDRESS.IsCurrent
    FROM ADDRESS
        JOIN CUSTOMERADDRESS
        ON CUSTOMERADDRESS.AddressID = ADDRESS.AddressID
    WHERE CUSTOMERADDRESS.IsCurrent > 0
    ) AS C_ADDRESS
    RIGHT JOIN CUSTOMER
    ON C_ADDRESS.CustomerID = CUSTOMER.CustomerID
WHERE City IN ('Winnipeg', 'London')
UNION
SELECT
    EMPLOYEE.FirstName AS FirstName,
    EMPLOYEE.LastName AS LastName,
    E_ADDRESS.City AS City
FROM
    (SELECT
        EMPLOYEEADDRESS.EmployeeID,
        ADDRESS.City,
        EMPLOYEEADDRESS.IsCurrent
    FROM ADDRESS
        JOIN EMPLOYEEADDRESS
        ON EMPLOYEEADDRESS.AddressID = ADDRESS.AddressID
    WHERE EMPLOYEEADDRESS.IsCurrent > 0
    ) AS E_ADDRESS
    RIGHT JOIN EMPLOYEE
    ON E_ADDRESS.EmployeeID = EMPLOYEE.EmployeeID
WHERE City IN ('Winnipeg', 'London');
```

Differences in SQL Syntax

- MySQL
  - No change
- SQL Server
  - No change
- Oracle
  - Replace ADDRESS with ASSIGN_1.ADDRESS on lines 10 and 28
  - Replace CUSTOMERADDRESS with ASSIGN_1.CUSTOMERADDRESS on line 11
  - Replace CUSTOMER with ASSIGN_1.CUSTOMER on line 15
  - Replace EMPLOYEEADDRESS with ASSIGN_1.EMPLOYEEADDRESS on line 29
  - Replace EMPLOYEE with ASSIGN_1.EMPLOYEE on line 33
  - Remove AS keyword from lines 14 and 32

Show which customers and employees are from each city

```
SELECT
    CUSTOMER.FirstName AS FirstName,
    CUSTOMER.LastName AS FirstName,
    C_ADDRESS.City AS City
FROM
    (SELECT
        CUSTOMERADDRESS.CustomerID,
        ADDRESS.City,
        CUSTOMERADDRESS.IsCurrent
    FROM ADDRESS
        JOIN CUSTOMERADDRESS
        ON CUSTOMERADDRESS.AddressID = ADDRESS.AddressID
    WHERE CUSTOMERADDRESS.IsCurrent > 0
    ) AS C_ADDRESS
    RIGHT JOIN CUSTOMER
    ON C_ADDRESS.CustomerID = CUSTOMER.CustomerID
WHERE City IS NOT NULL
UNION
SELECT
    EMPLOYEE.FirstName AS FirstName,
    EMPLOYEE.LastName AS LastName,
     E_ADDRESS.City AS City
FROM
    (SELECT
        EMPLOYEEADDRESS.EmployeeID,
        ADDRESS.City,
        EMPLOYEEADDRESS.IsCurrent
    FROM ADDRESS
        JOIN EMPLOYEEADDRESS
        ON EMPLOYEEADDRESS.AddressID = ADDRESS.AddressID
    WHERE EMPLOYEEADDRESS.IsCurrent > 0
    ) AS E_ADDRESS
    RIGHT JOIN EMPLOYEE
    ON E_ADDRESS.EmployeeID = EMPLOYEE.EmployeeID
WHERE City IS NOT NULL;
```

Differences in SQL Syntax

- MySQL
  - No change
- SQL Server
  - No change
- Oracle
  - Replace ADDRESS with ASSIGN_1.ADDRESS on lines 10 and 28
  - Replace CUSTOMERADDRESS with ASSIGN_1.CUSTOMERADDRESS on line 11
  - Replace CUSTOMER with ASSIGN_1.CUSTOMER on line 15
  - Replace EMPLOYEEADDRESS with ASSIGN_1.EMPLOYEEADDRESS on line 29
  - Replace EMPLOYEE with ASSIGN_1.EMPLOYEE on line 33
  - Remove AS keyword from lines 14 and 32

Display the employees who are also customers

```
SELECT
    emp.CustomerID,
    emp.EmployeeID,
    emp.FirstName,
    emp.LastName,
    emp.Notes
FROM EMPLOYEE
    JOIN
        (SELECT
            EMPLOYEECUSTOMER.EmployeeID,
            CUSTOMER.CustomerID,
            CUSTOMER.FirstName,
            CUSTOMER.LastName,
            CUSTOMER.Notes
        FROM CUSTOMER
            JOIN EMPLOYEECUSTOMER
            ON CUSTOMER.CustomerID=EMPLOYEECUSTOMER.CustomerID
        ) AS emp
    ON EMPLOYEE.EmployeeID=emp.EmployeeID;
```

Differences in SQL Syntax

- MySQL
  - No change
- SQL Server
  - No change
- Oracle
  - Replace EMPLOYEE with ASSIGN_1.EMPLOYEE on line 7
  - Replace CUSTOMER with ASSIGN_1.CUSTOMER on line 15
  - Replace EMPLOYEECUSTOMER with ASSIGN_1.EMPLOYEECUSTOMER on line 16
  - Remove AS keyword from line 18

Show all employees who are customers if their last name starts with M or contains a C

```
SELECT
    emp.CustomerID,
    emp.EmployeeID,
    emp.FirstName,
    emp.LastName,
    emp.Notes
FROM EMPLOYEE
    JOIN
        (SELECT
            EMPLOYEECUSTOMER.EmployeeID,
            CUSTOMER.CustomerID,
            CUSTOMER.FirstName,
            CUSTOMER.LastName,
            CUSTOMER.Notes
        FROM CUSTOMER
            JOIN EMPLOYEECUSTOMER
            ON CUSTOMER.CustomerID=EMPLOYEECUSTOMER.CustomerID
        ) AS emp
    ON EMPLOYEE.EmployeeID=emp.EmployeeID
WHERE
    emp.LastName LIKE 'm%'
    OR emp.LastName LIKE '%c%';
```

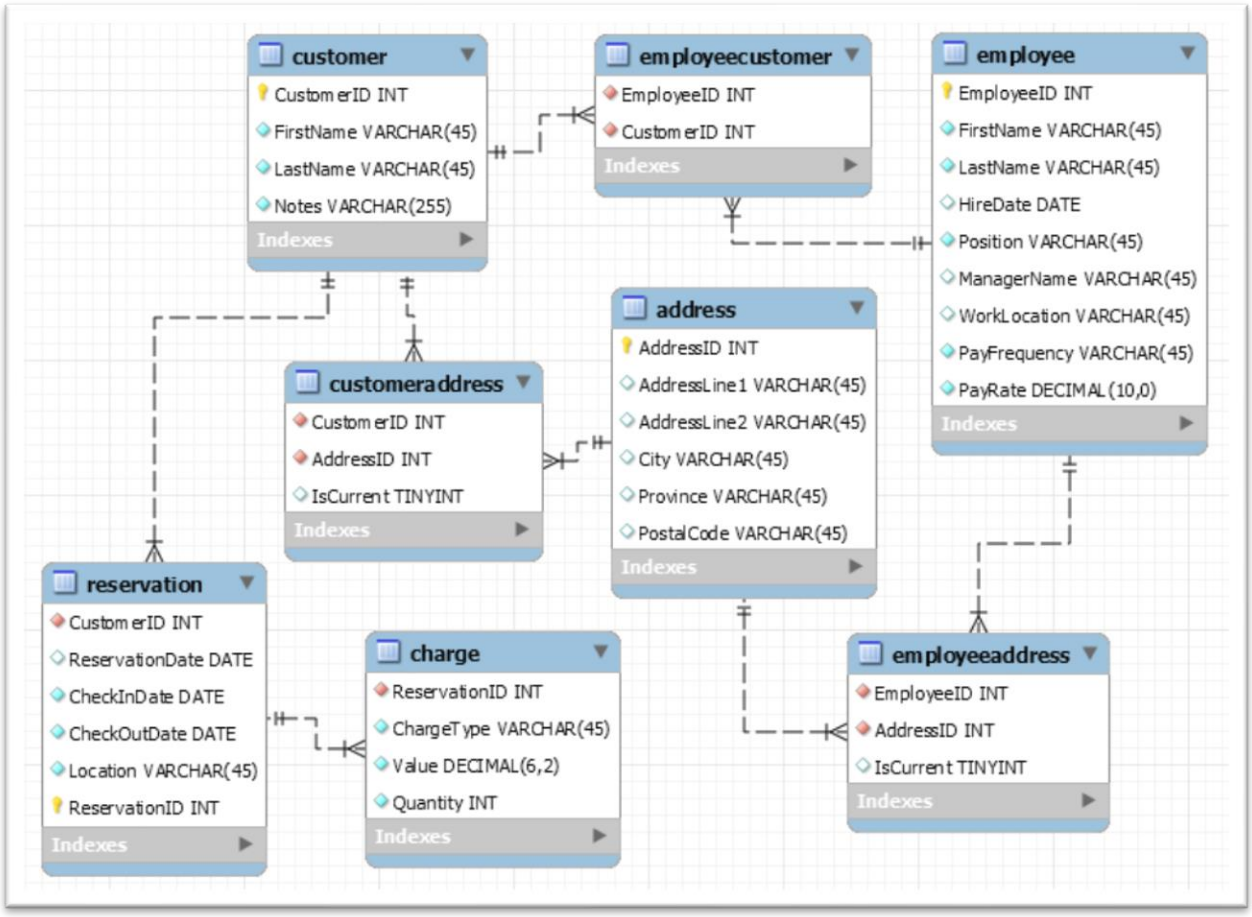
Differences in SQL Syntax

- MySQL
  - No change
- SQL Server
  - No change
- Oracle
  - Replace EMPLOYEE with ASSIGN_1.EMPLOYEE on line 7
  - Replace CUSTOMER with ASSIGN_1.CUSTOMER on line 15
  - Replace EMPLOYEECUSTOMER with ASSIGN_1.EMPLOYEECUSTOMER on line 16
  - Remove AS keyword from line 18

Databases

MySQL

MySQL Database Design Diagram



MySQL Table Fields

ADDRESS Table

Column	Type	Default ...	Nullable
AddressID	int		NO
AddressLine1	varchar(45)		YES
AddressLine2	varchar(45)		YES
City	varchar(45)		YES
PostalCode	varchar(45)		YES
Province	varchar(45)		YES

CHARGE Table

Column	Type	Default Value	Nullable
ChargeType	varchar(45)		NO
Quantity	int	1	NO
ReservationID	int		NO
Value	decimal(6,2)	0.00	NO

CUSTOMER Table

Column	Type	Default Value	Nullable
CustomerID	int		NO
FirstName	varchar(45)		NO
LastName	varchar(45)		NO
Notes	varchar(255)	Use regular rates	NO

CUSTOMERADDRESS Table

Column	Type	Default Value	Nullable
AddressID	int		NO
CustomerID	int		NO
IsCurrent	tinyint	1	YES

EMPLOYEE Table

Column	Type	Default Value	Nullable
EmployeeID	int		NO
FirstName	varchar(45)		NO
HireDate	date		YES
LastName	varchar(45)		NO
ManagerID	int		YES
PayFrequency	varchar(45)	Hourly	NO
PayRate	decimal(10,0)	0	NO
Position	varchar(45)		NO
WorkLocation	varchar(45)		YES

EMPLOYEEADDRESS Table

Column	Type	Default Value	Nullable
AddressID	int		NO
EmployeeID	int		NO
IsCurrent	tinyint	1	YES

EMPLOYEECUSTOMER Table

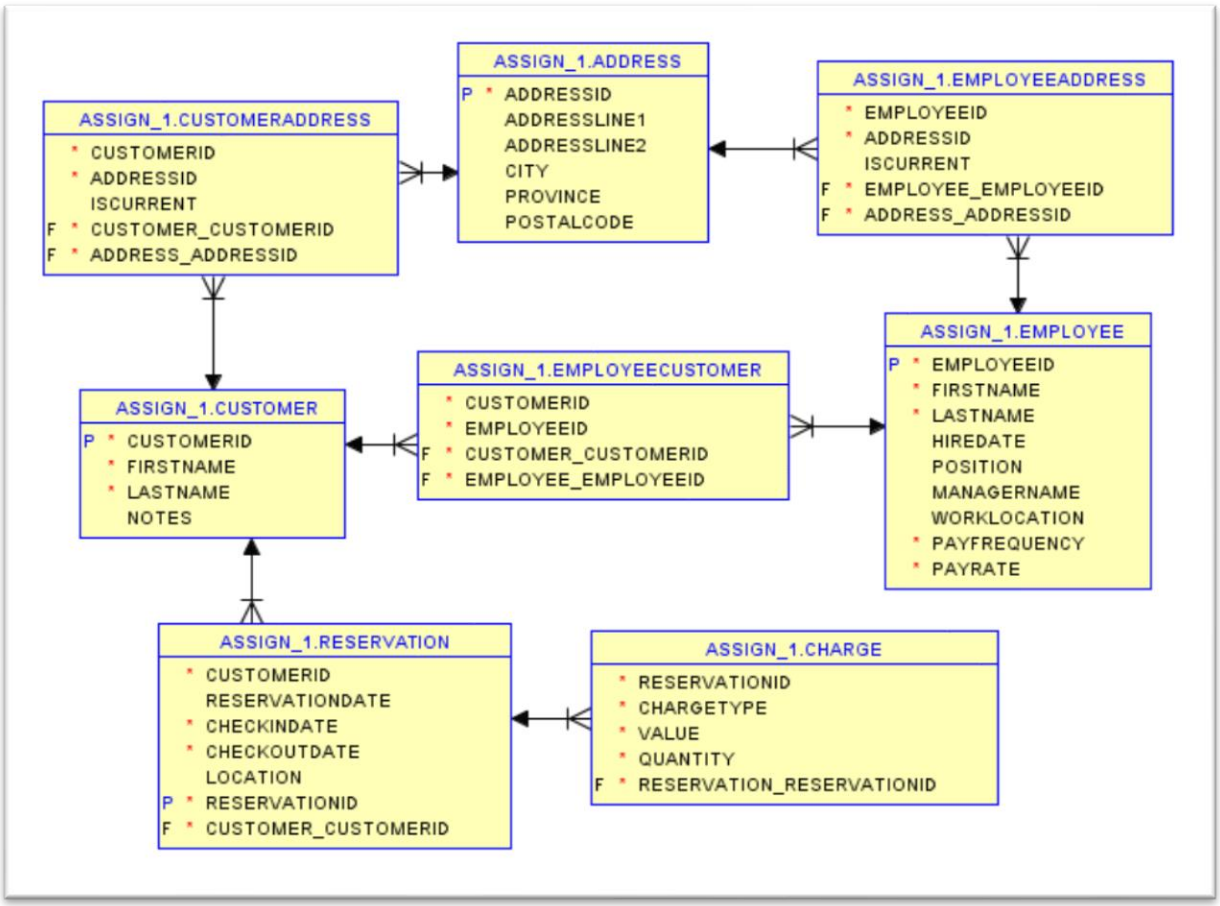
Column	Type	Default Value	Nullable
CustomerID	int		NO
EmployeeID	int		NO

RESERVATION Table

Column	Type	Default Value	Nullable
CheckInDate	date		NO
CheckOutDate	date		NO
CustomerID	int		NO
Location	varchar(45)		NO
ReservationDate	date		YES
ReservationID	int		NO

Oracle Database

Oracle Database Design Diagram



Oracle Table Fields

ADDRESS Table

	COLUMN_NAME	DATA_TYPE	NULLABLE
1	ADDRESSID	NUMBER(38,0)	No
2	ADDRESSLINE1	VARCHAR2(45 BYTE)	Yes
3	ADDRESSLINE2	VARCHAR2(45 BYTE)	Yes
4	CITY	VARCHAR2(45 BYTE)	Yes
5	PROVINCE	VARCHAR2(45 BYTE)	Yes
6	POSTALCODE	VARCHAR2(6 BYTE)	Yes

CHARGE Table

	COLUMN_NAME	DATA_TYPE	NULLABLE
1	RESERVATIONID	NUMBER(38,0)	No
2	CHARGETYPE	VARCHAR2(45 BYTE)	No
3	VALUE	NUMBER(6,2)	No
4	QUANTITY	NUMBER(38,0)	No

CUSTOMER Table

	COLUMN_NAME	DATA_TYPE	NULLABLE
1	CUSTOMERID	NUMBER(38,0)	No
2	FIRSTNAME	VARCHAR2(45 BYTE)	No
3	LASTNAME	VARCHAR2(45 BYTE)	No
4	NOTES	VARCHAR2(255 BYTE)	Yes

CUSTOMERADDRESS Table

	COLUMN_NAME	DATA_TYPE	NULLABLE
1	CUSTOMERID	NUMBER(38,0)	No
2	ADDRESSID	NUMBER(38,0)	No
3	ISCURRENT	NUMBER(38,0)	Yes

EMPLOYEE Table

	COLUMN_NAME	DATA_TYPE	NULLABLE
1	EMPLOYEEID	NUMBER(38,0)	No
2	FIRSTNAME	VARCHAR2(45 BYTE)	No
3	LASTNAME	VARCHAR2(45 BYTE)	No
4	HIREDATE	DATE	Yes
5	POSITION	VARCHAR2(45 BYTE)	Yes
6	MANAGERID	NUMBER(38,0)	Yes
7	WORKLOCATION	VARCHAR2(45 BYTE)	Yes
8	PAYFREQUENCY	VARCHAR2(45 BYTE)	No
9	PAYRATE	NUMBER(12,2)	No

EMPLOYEEADDRESS Table

	COLUMN_NAME	DATA_TYPE	NULLABLE
1	EMPLOYEEID	NUMBER(38,0)	No
2	ADDRESSID	NUMBER(38,0)	No
3	ISCURRENT	NUMBER(38,0)	Yes

EMPLOYEECUSTOMER Table

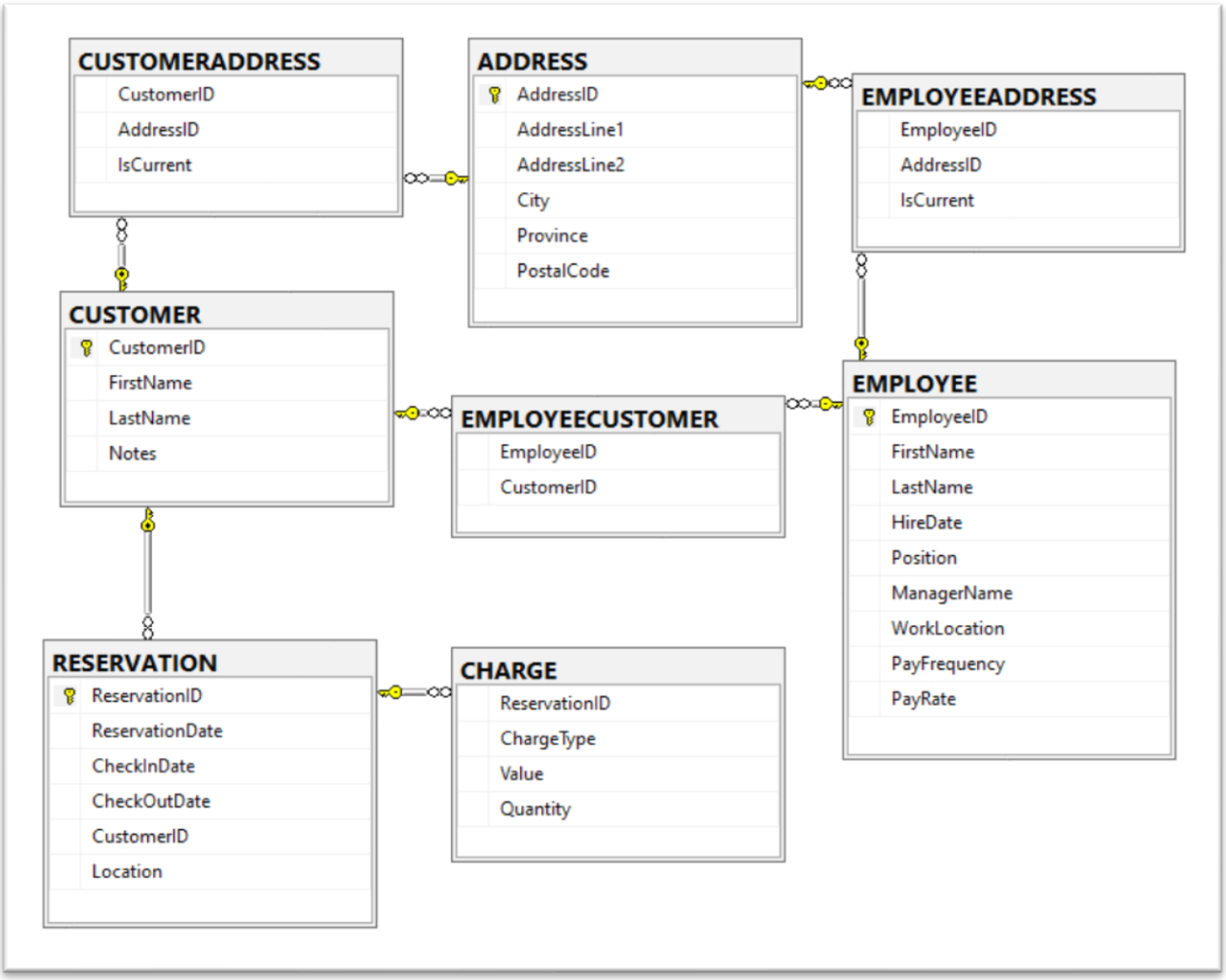
	COLUMN_NAME	DATA_TYPE	NULLABLE
1	CUSTOMERID	NUMBER(38,0)	No
2	EMPLOYEEID	NUMBER(38,0)	No

RESERVATION Table

	COLUMN_NAME	DATA_TYPE	NULLABLE
1	CUSTOMERID	NUMBER(38,0)	No
2	RESERVATIONDATE	DATE	Yes
3	CHECKINDATE	DATE	No
4	CHECKOUTDATE	DATE	No
5	LOCATION	VARCHAR2(45 BYTE)	Yes
6	RESERVATIONID	NUMBER(38,0)	No

SQL Server

SQL Server Database Design Diagram



SQL Server Table Fields

ADDRESS Table

Column Name	Data Type	Allow Nulls
AddressID	int	☐
AddressLine1	varchar(45)	☑
AddressLine2	varchar(45)	☑
City	varchar(45)	☑
Province	varchar(45)	☑
PostalCode	char(6)	☑

CHARGE Table

Column Name	Data Type	Allow Nulls
ReservationID	int	☐
ChargeType	varchar(45)	☐
Value	decimal(6, 2)	☐
Quantity	int	☐

CUSTOMER Table

Column Name	Data Type	Allow Nulls
CustomerID	int	☐
FirstName	varchar(45)	☐
LastName	varchar(45)	☐
Notes	varchar(255)	☑

CUSTOMERADDRESS Table

Column Name	Data Type	Allow Nulls
CustomerID	int	☐
AddressID	int	☐
IsCurrent	tinyint	☑

EMPLOYEE Table

Column Name	Data Type	Allow Nulls
EmployeeID	int	☐
FirstName	varchar(45)	☐
LastName	varchar(45)	☐
HireDate	date	☑
Position	varchar(45)	☑
ManagerID	int	☑
WorkLocation	varchar(45)	☑
PayFrequency	varchar(45)	☐
PayRate	decimal(10, 2)	☐

EMPLOYEEADDRESS Table

Column Name	Data Type	Allow Nulls
EmployeeID	int	☐
AddressID	int	☐
IsCurrent	tinyint	☐

EMPLOYEECUSTOMER Table

Column Name	Data Type	Allow Nulls
EmployeeID	int	☐
CustomerID	int	☐

RESERVATION Table

Column Name	Data Type	Allow Nulls
ReservationID	int	☐
ReservationDate	date	☑
CheckInDate	date	☐
CheckOutDate	date	☐
CustomerID	int	☐
Location	varchar(45)	☑

Appendix: Assumptions and Notes

- Started with MySQL, since it's the only DBMS I have experience with
- I tried to make the table names all caps, but MySQL wouldn't allow capitals in table names
- MySQL wouldn't let me use BOOLEAN datatype for the IsCurrent columns in the CUSTOMERADDRESS and EMPLOYEEADDRESS tables, so I used TINYINT datatype instead
- ADDRESS table contains more fields than strictly needed for the information given, but for robust design and expandability Van Whinkle should be gathering full addresses for their customers and employees rather than just cities
- By the same token, there should be more contact info for both customers and employees that I didn't include, such as Phone Number and Email Address
- Eventually Van Whinkle would need to start collecting billing info for customers and direct deposit info for employees
- To reduce repetition, I referenced the same Toronto record in the ADDRESS table for both Bob Smith (desk clerk) and Billy Elliot, even though their addresses would be different in reality
- Neither BOOLEAN nor TINYINT were options in Oracle, so I used SMALLINT in hopes that it would be close enough
- Changed EMPLOYEE.ManagerName to EMPLOYEE.ManagerID after diagrams were complete, I updated the table and data screenshots but not the diagrams
- Did not set default values for columns in Oracle or SQL Server