Carlton B. Ramsey
@eccentricDBA
www.eccentricDBA.com

T-SQL Basics

About Me

- Application Developer turned Database Administrator
- MCSA: SQL 2016 Database Development
- Past Chapter President / Chapter Board of Director Member Akron-Canton Chapter of the CompTIA Association of Information Technology Professionals
- Civic Hacker

About You

Each and every one of you have something that you can teach each and every one of us.

~Allen White @SQLRunr

Our Agenda

- Basic Database Structures
- Retrieving Records
- Adding Records
- Removing Records
- Changing Records
- Advance Query Techniques
- Joins
- Demo

Basic Database Structure

- Database
  - Table
    - Columns
    - Indexes

Basic Database Structure - Database

https://flic.kr/p/7m6Uqz

Basic Database Structure - Table

https://flic.kr/p/7m6Uri

Basic Database Structure - Indexes

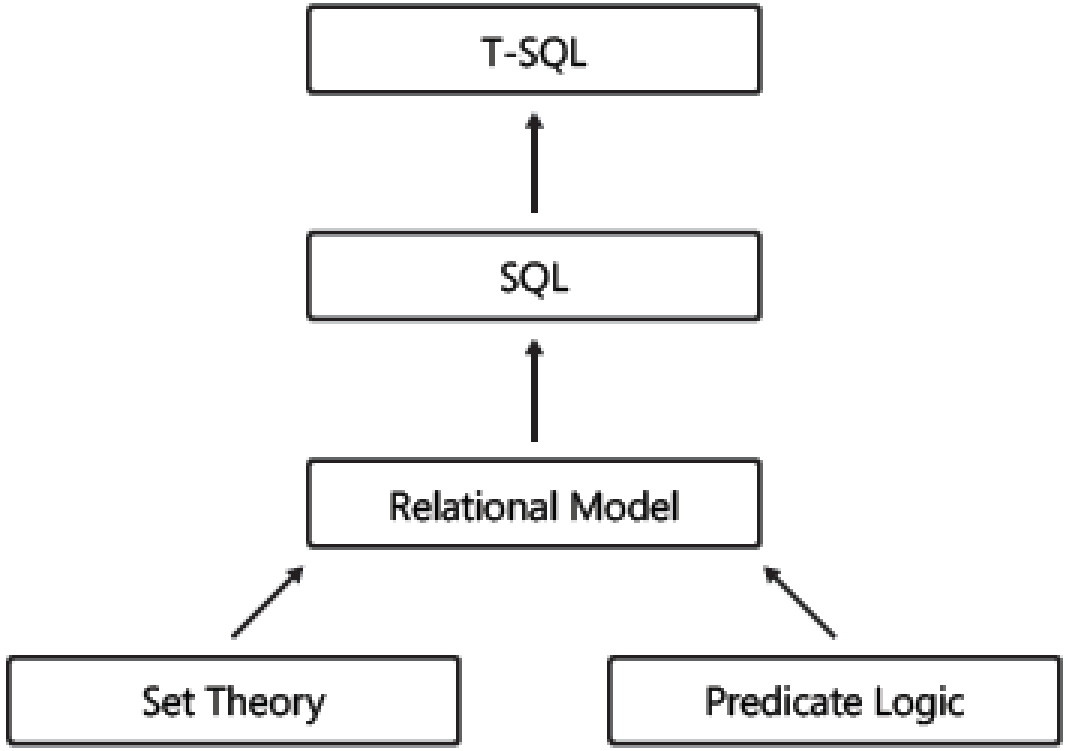
https://flic.kr/p/7fnGQC

What is T-SQL

T-SQL Transact-SQL is a Microsoft and Sybase extension of SQL. SQL is an acronym for Structured Query Language.

It is the language of the database that enables you to communicate with the database engine.

Evolution of T-SQL



https://www.microsoftpressstore.com/articles/article.aspx?p=2201633&seqNum=2

Set Theory

https://www.elcsd.org/cms/lib/NY01000534/Centricity/Domain/84/venn_3.gif

Predicate Logic

Predicates

p: GB is a country

Country(great_britain)

q: John loves Mary

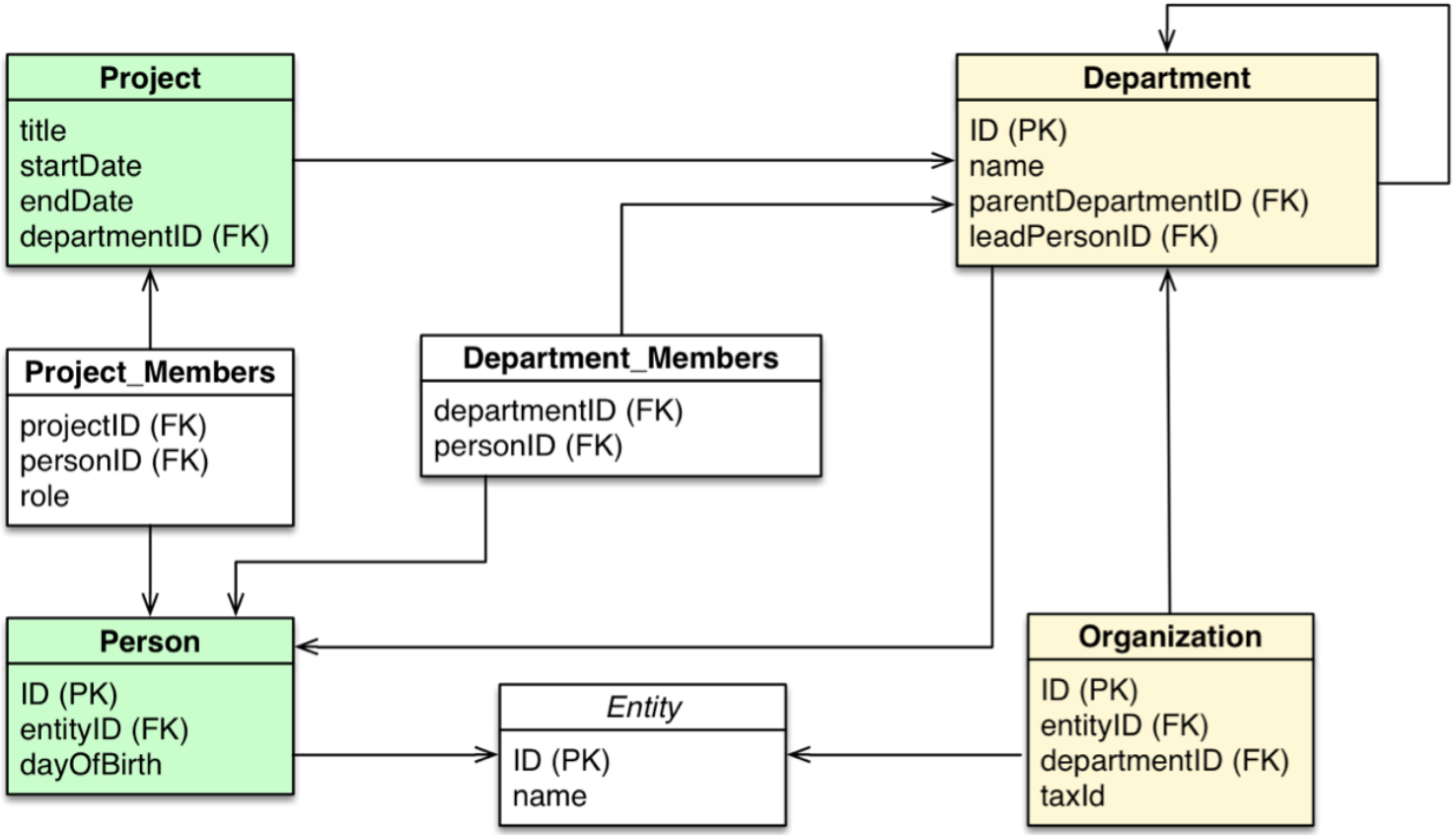
Love(john,mary)

r: Jane sends Paul a letter.

Send(jane,paul,letter)

https://www.youtube.com/watch?v=IhodKMPwShc

Relational Model



https://s3.amazonaws.com/dev.assets.neo4j.com/wp-content/uploads/20160229120043/organization-relational-model.png

Retrieving Records

SELECT

FROM

WHERE

Retrieving Records

```
SELECT * FROM [dbo].[ParentTable] WHERE [Value] = 'Movies'
```

100 %

Results | Messages

	ID	Value
1	1	Movies

How SQL is written vs processed

Keyed-in Order

SELECT

FROM

WHERE

GROUP BY

HAVING

ORDER BY

Logical Query Processing Order

FROM

WHERE

GROUP BY

HAVING

SELECT

ORDER BY

Adding Records

INSERT INTO VALUES

Adding Records

```
INSERT INTO [dbo].[ParentTable]
            ([Value])
      VALUES
            ('Movies')
GO
```

100 %

Messages

```
(1 row(s) affected)
```

Removing Records

DELETE FROM WHERE

Removing Records

```
SELECT * FROM [dbo].[ParentTable] WHERE [Value] = 'Movies'
```

100 %

Results | Messages

	ID	Value
1	1	Movies

```
DELETE FROM [dbo].[ParentTable] WHERE ID = 1
```

100 %

Messages

```
(1 row(s) affected)
```

Changing Records

UPDATE

SET

WHERE

Changing Records

```sql
SELECT * FROM [dbo].[ParentTable]

UPDATE [dbo].[ParentTable]
   SET [Value] = 'Top Movies'
 WHERE [ID] = 1

SELECT * FROM [dbo].[ParentTable]
```

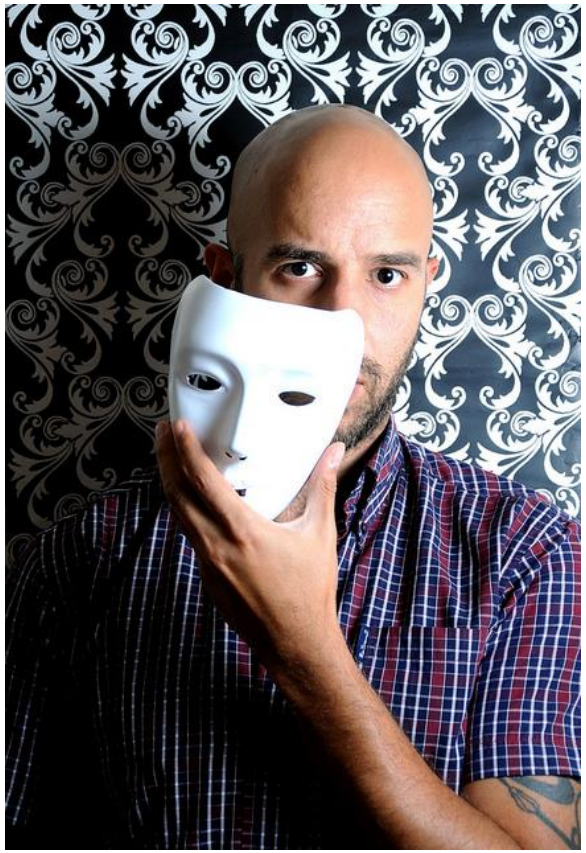
100 %

Results

```
ID          Value
----------- -------------------------
1           Movies

(1 row(s) affected)

(1 row(s) affected)

ID          Value
----------- -------------------------
1           Top Movies

(1 row(s) affected)
```

Advance Query Techniques – Calculations and Aliases

AS



https://flic.kr/p/8rKioi

Advance Query Techniques – Calculations and Aliases

```
SELECT 1 AS [Value]
```

100 %

Results | Messages

	Value
1	1

```
SELECT 1 + 1 AS [Value]
```

100 %

Results | Messages

	Value
1	2

Advance Query Techniques – Functions

LEFT

RIGHT

SUBSTRING

LTRIM

RTRIM

UPPER

LOWER

GETDATE

Advance Query Techniques – Functions

```sql
SELECT Value
       , LEFT(Value, 1) AS [LEFT]
       , RIGHT(Value, 1) AS [RIGHT]
       , SUBSTRING(Value, 0, 4) AS [SUBSTRING]
       , ']' + LTRIM('      123      ') + '[' AS [LTRIM]
       , ']' + RTRIM('      123      ') + '[' AS [RTRIM]
       , UPPER(Value) AS [UPPER]
       , LOWER(Value) AS [LOWER]
       , GETDATE() AS [GETDATE]
  FROM [dbo].[ParentTable]
 WHERE ID = 1
```

100 %

Results | Messages

	Value	LEFT	RIGHT	SUBSTRING	LTRIM	RTRIM	UPPER	LOWER	GETDATE
1	Top Movies	T	s	Top	]123 [	] 123[	TOP MOVIES	top movies	2014-06-09 20:41:40.770

Advance Query Techniques – Sorting

ORDER BY

ASC

DESC

Advance Query Techniques – Sorting

```sql
SELECT name
  FROM sys.tables
ORDER BY name

SELECT name
  FROM sys.tables
ORDER BY name DESC
```

100 %

Results | Messages

	name
1	MSreplication_options
2	spt_fallback_db
3	spt_fallback_dev
4	spt_fallback_usg
5	spt_monitor

	name
1	spt_monitor
2	spt_fallback_usg
3	spt_fallback_dev
4	spt_fallback_db
5	MSreplication_options

Advance Query Techniques – Column-Based Logic

CASE

WHEN

THEN

ELSE

END

Advance Query Techniques – Column-Based Logic

```sql
SELECT name
       ,uses_ansi_nulls
       ,CASE uses_ansi_nulls
            WHEN 1 THEN 'Yes'
            WHEN 0 THEN 'No'
            ELSE 'Unkown'
        END AS uses_ansi_nulls_desc
  FROM sys.tables
```

100 %

Results | Messages

	name	uses_ansi_nulls	uses_ansi_nulls_desc
1	spt_fallback_db	1	Yes
2	spt_fallback_dev	1	Yes
3	spt_fallback_usg	1	Yes
4	MSreplication_options	0	No
5	spt_monitor	0	No

Advance Query Techniques – Row-Based Logic

WHERE

TOP

Advance Query Techniques – Row-Based Logic

```sql
SELECT  TOP 2
        name
       ,uses_ansi_nulls
  FROM sys.tables
 WHERE uses_ansi_nulls = 1
```

100 %

Results | Messages

	name	uses_ansi_nulls
1	spt_fallback_db	1
2	spt_fallback_dev	1

Advance Query Techniques – Boolean Logic

AND

OR

NOT

BETWEEN

IN

IS

NULL

Advance Query Techniques – Boolean Logic

```sql
SELECT 'Selected' WHERE 1 = 1 AND 2 = 2
SELECT 'Selected' WHERE 1 = 1 AND 2 = 3
SELECT 'Selected' WHERE 1 = 1 OR 2 = 3
SELECT 'Selected' WHERE NOT (1 = 1) OR 2 = 3
```

100 %

Results | Messages

	(No column name)
1	Selected

(No column name)

	(No column name)
1	Selected

(No column name)

Advance Query Techniques – Boolean Logic

```sql
SELECT 'Selected' WHERE 1 BETWEEN 1 AND 10
SELECT 'Selected' WHERE 1 NOT BETWEEN 1 AND 10
SELECT 'Selected' WHERE 1 IN (1, 2,10,20)
SELECT 'Selected' WHERE 1 NOT IN (1, 2,10,20)
```

100 %

Results | Messages

	(No column name)
1	Selected

(No column name)

	(No column name)
1	Selected

(No column name)

Advance Query Techniques – Boolean Logic

```sql
DECLARE @i INT
SELECT 'Selected' WHERE @i IS NULL
SELECT 'Selected' WHERE @i IS NOT NULL
```

100 %

Results | Messages

	(No column name)
1	Selected

(No column name)

Advance Query Techniques – Inexact Matches

LIKE

Advance Query Techniques – Inexact Matches

```sql
DECLARE @Text VARCHAR(100)
SET @Text = 'I love SQLSaturday'
SELECT 'Selected' WHERE @Text LIKE '%SQL'
SELECT 'Selected' WHERE @Text LIKE '%SQL%'
SELECT 'Selected' WHERE @Text LIKE '% SQL%'
SELECT 'Selected' WHERE @Text LIKE '%SQL %'
```

100 %

Results | Messages

	(No column name)

	(No column name)
1	Selected

	(No column name)
1	Selected

	(No column name)

Advance Query Techniques – Summarizing Records

DISTINCT

MAX

SUM

COUNT

AVG

GROUP BY

MIN

HAVING

Advance Query Techniques – Summarizing Records

```sql
DECLARE @Table AS TABLE (Name VARCHAR(200),Salary Money,IsManager BIT)

INSERT INTO @Table (Name, Salary, IsManager) VALUES ('John Smith', 10000.00, 0)
INSERT INTO @Table (Name, Salary, IsManager) VALUES ('Jane Doe', 10500.00, 0)
INSERT INTO @Table (Name, Salary, IsManager) VALUES ('Bob Gates', 10500.00, 1)
INSERT INTO @Table (Name, Salary, IsManager) VALUES ('Jane Doe', 10500.00, 0)

SELECT * FROM @Table
```

100 %

Results | Messages

	Name	Salary	IsManager
1	John Smith	10000.00	0
2	Jane Doe	10500.00	0
3	Bob Gates	10500.00	1
4	Jane Doe	10500.00	0

Advance Query Techniques – Summarizing Records

```sql
SELECT * FROM @Table

SELECT DISTINCT * FROM @Table
```

100 %

Results | Messages

	Name	Salary	IsManager
1	John Smith	10000.00	0
2	Jane Doe	10500.00	0
3	Bob Gates	10500.00	1
4	Jane Doe	10500.00	0

	Name	Salary	IsManager
1	Bob Gates	10500.00	1
2	Jane Doe	10500.00	0
3	John Smith	10000.00	0

Advance Query Techniques – Summarizing Records

```
SELECT SUM(Salary) AS [Sum], AVG(Salary) AS [Avg], MIN(Salary) AS [Min], MAX(Salary) AS [Max], COUNT(*) AS [Count]  FROM @Table
```

100 %

Results | Messages

	Sum	Avg	Min	Max	Count
1	41500.00	10375.00	10000.00	10500.00	4

Advance Query Techniques – Summarizing Records

```sql
SELECT IsManager, SUM(Salary) AS [Sum], AVG(Salary) AS [Avg], MIN(Salary) AS [Min], MAX(Salary) AS [Max], COUNT(*) AS [Count]  FROM @Table
```

100 %

Messages

```
Msg 8120, Level 16, State 1, Line 14
Column '@Table.IsManager' is invalid in the select list because it is not contained in either an aggregate function or the GROUP BY clause.
```

```sql
SELECT IsManager, SUM(Salary) AS [Sum], AVG(Salary) AS [Avg], MIN(Salary) AS [Min], MAX(Salary) AS [Max], COUNT(*) AS [Count]  FROM @Table GROUP BY IsManager
```

100 %

Results | Messages

	IsManager	Sum	Avg	Min	Max	Count
1	0	31000.00	10333.3333	10000.00	10500.00	3
2	1	10500.00	10500.00	10500.00	10500.00	1

Advance Query Techniques – Summarizing Records

```sql
SELECT Name, COUNT(*) FROM @table GROUP BY Name
SELECT Name, COUNT(*) FROM @table GROUP BY Name HAVING COUNT(*) > 1
```

100 %

Results | Messages

	Name	(No column name)
1	Bob Gates	1
2	Jane Doe	2
3	John Smith	1

	Name	(No column name)
1	Jane Doe	2

https://flic.kr/p/98RSdj

Set Operators and Joins

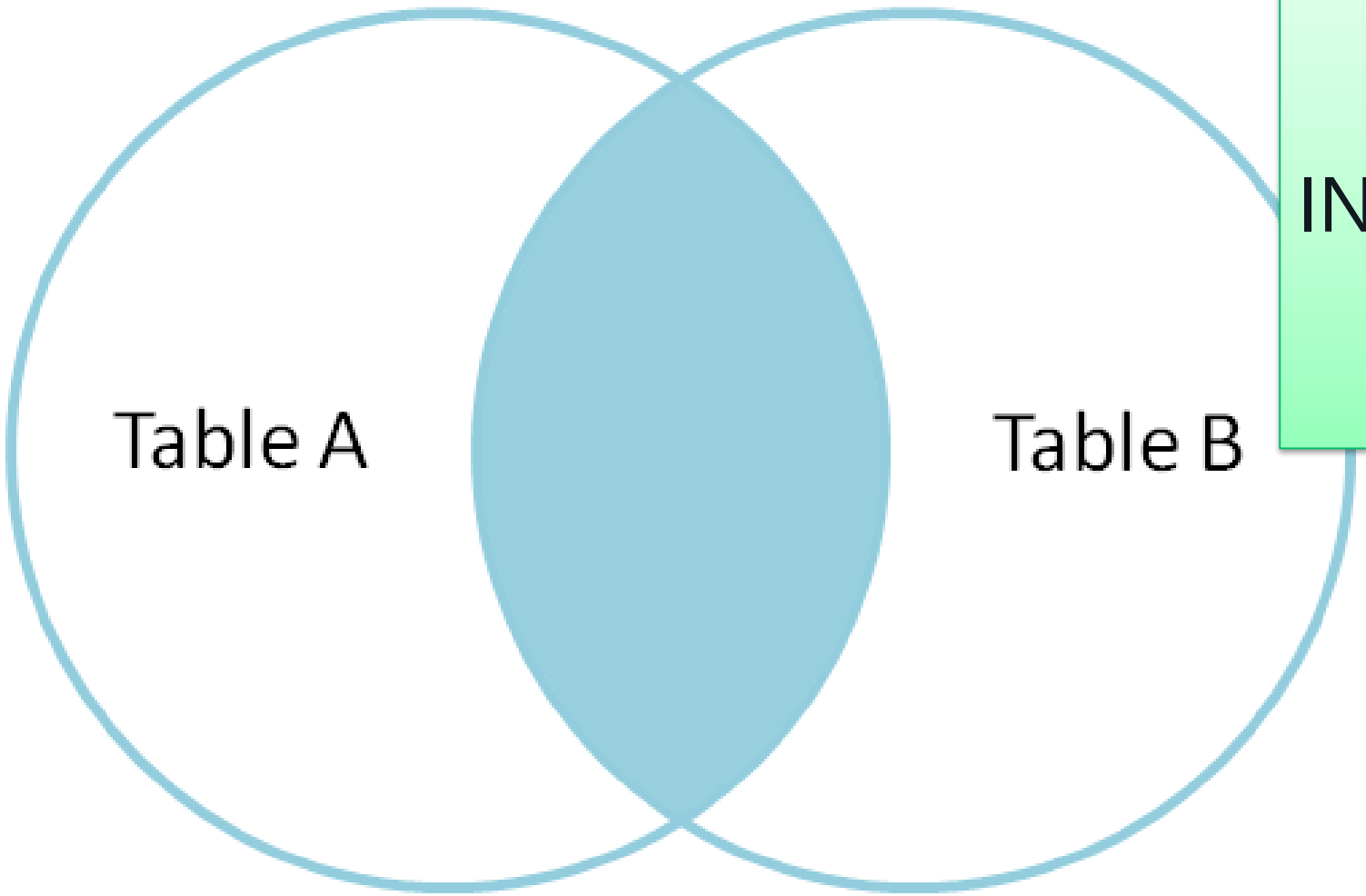
Set Operators

- INTERSECT
- EXCEPT
- UNION (ALL)

Joins

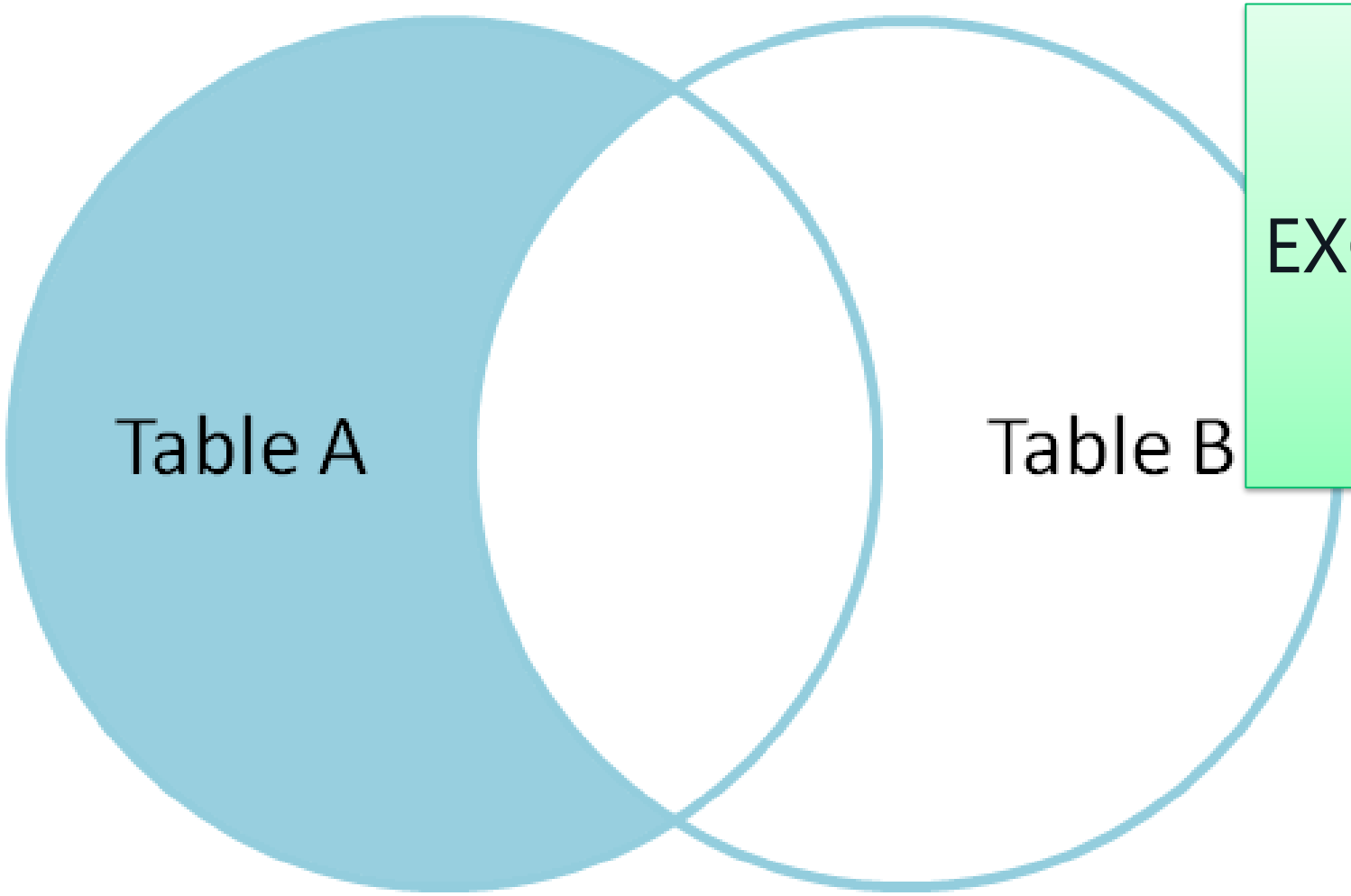
- INNER JOIN
- LEFT JOIN
- FULL OUTER JOIN

Set Operators – INTERSECT



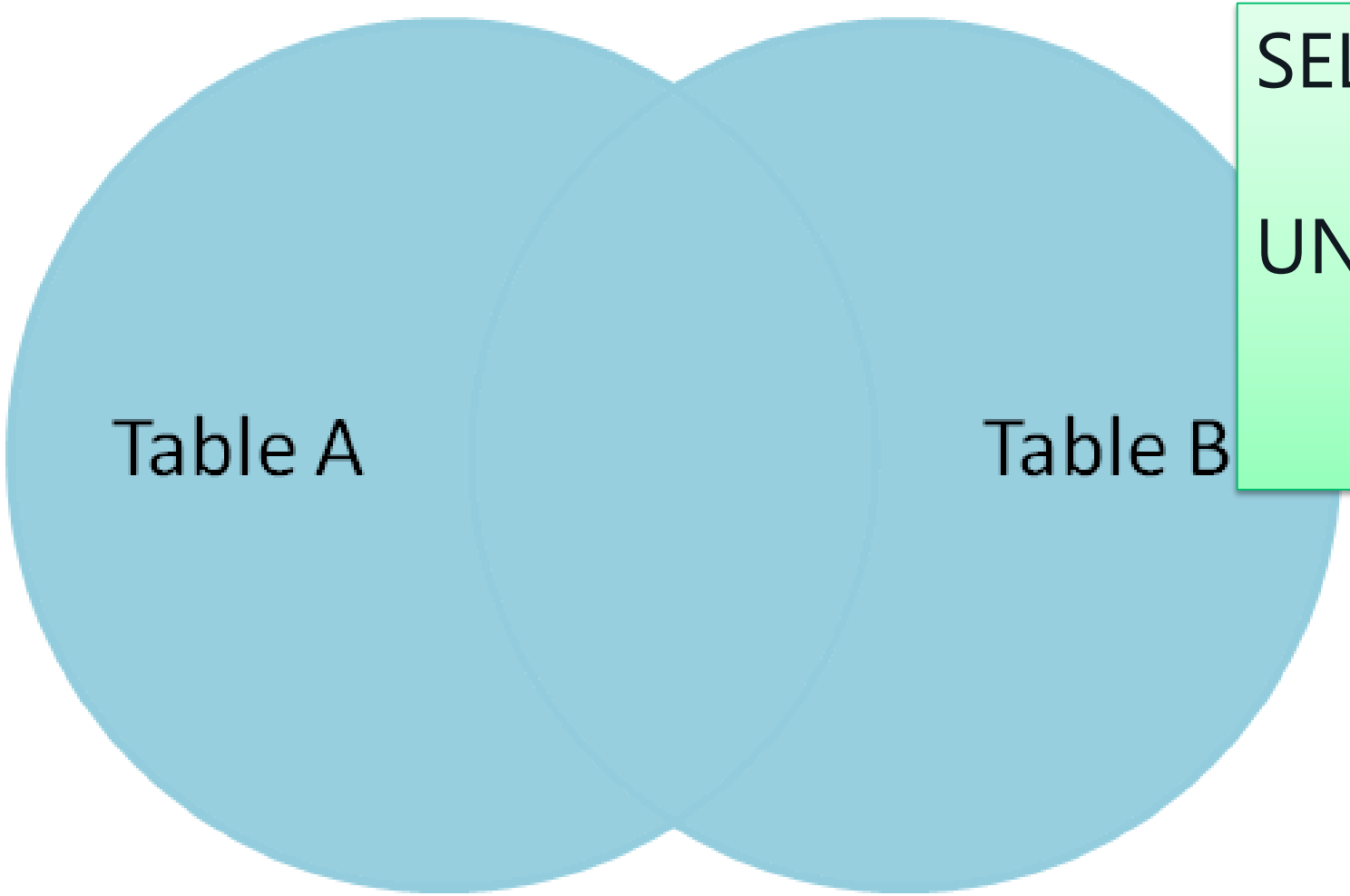

```
SELECT city
    FROM Employees
INTERSECT
    SELECT city
    FROM Customers
```

Set Operators – EXCEPT



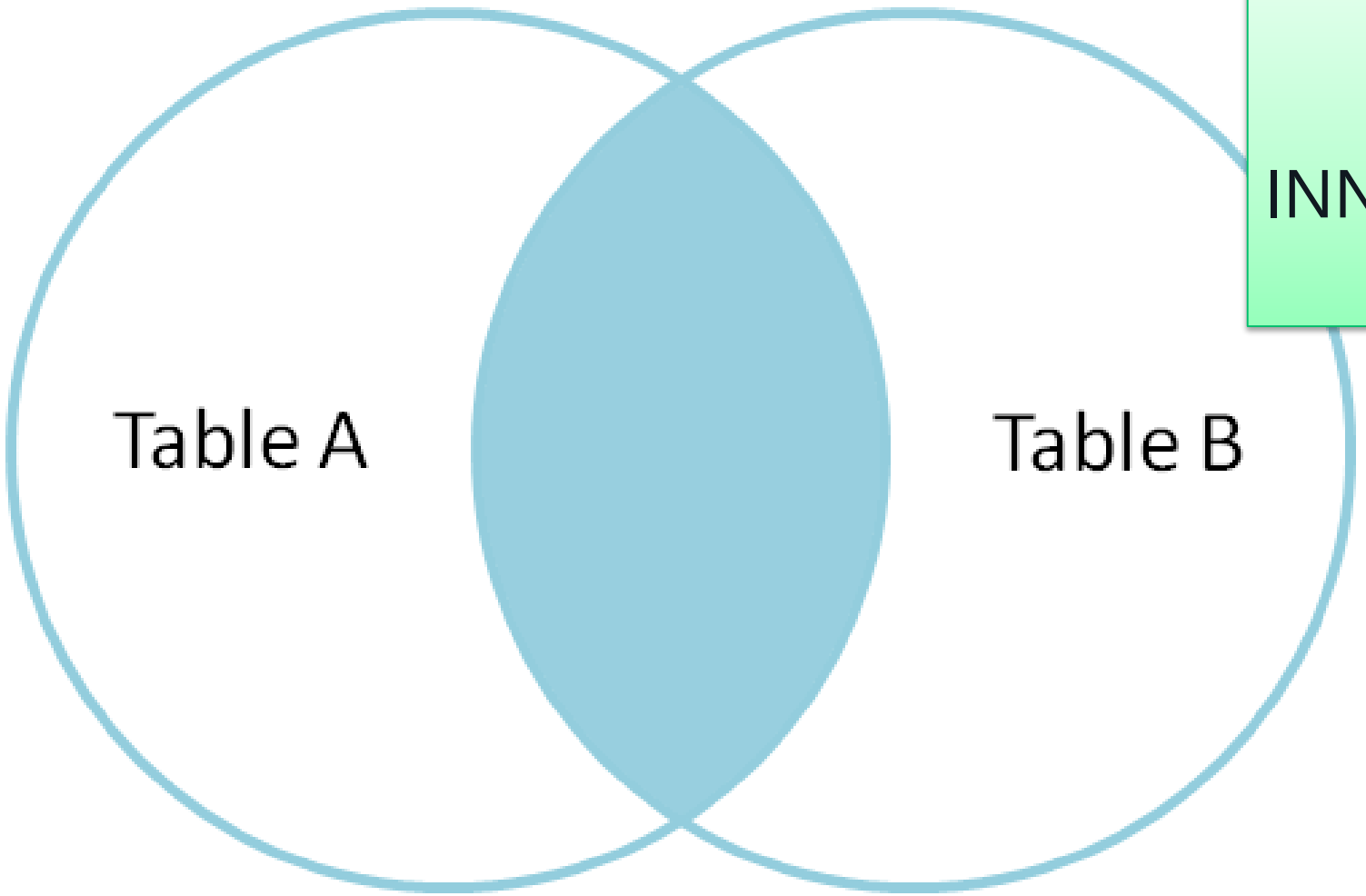

```
SELECT city
    FROM Employees
EXCEPT
    SELECT city
    FROM Customers
```

Set Operators – UNION (ALL)



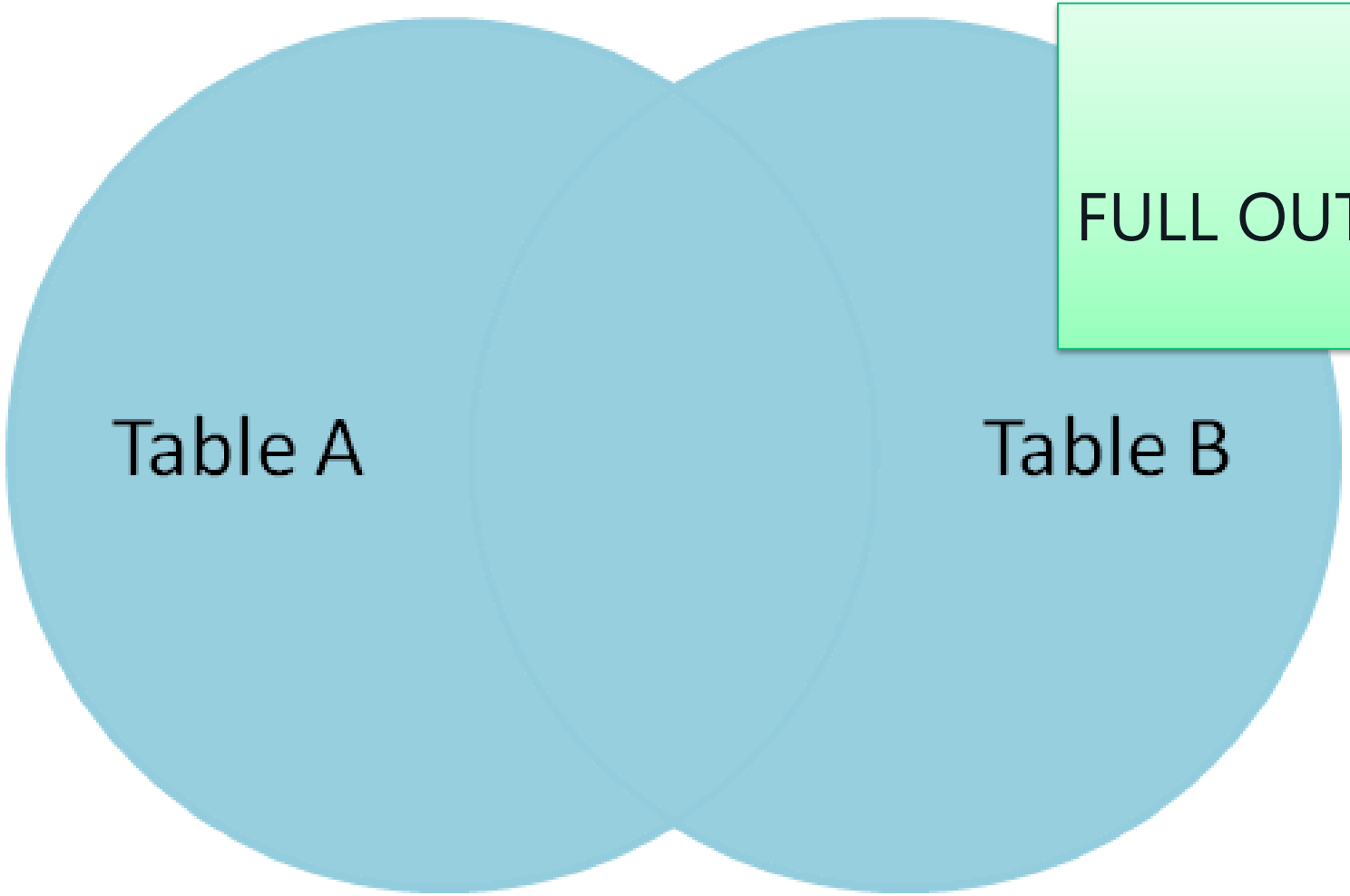

```
SELECT city
        FROM Employees
UNION ALL
      SELECT city
        FROM Customers
```

Joins – Inner Join



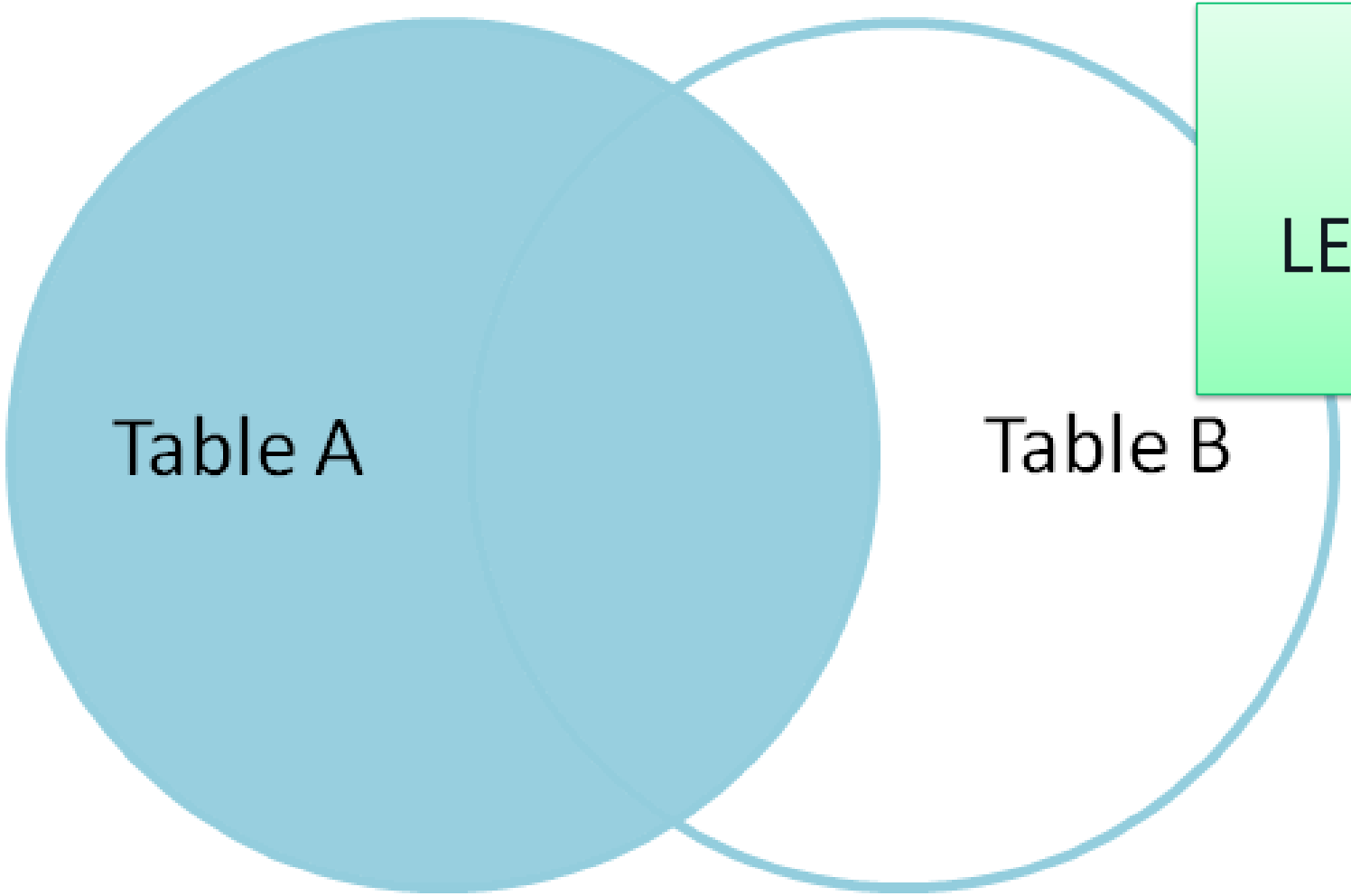

```
SELECT *
    FROM TableA AS A
INNER JOIN TableB AS B
        ON A.ID = B.ID
```

Joins – Full Outer Join



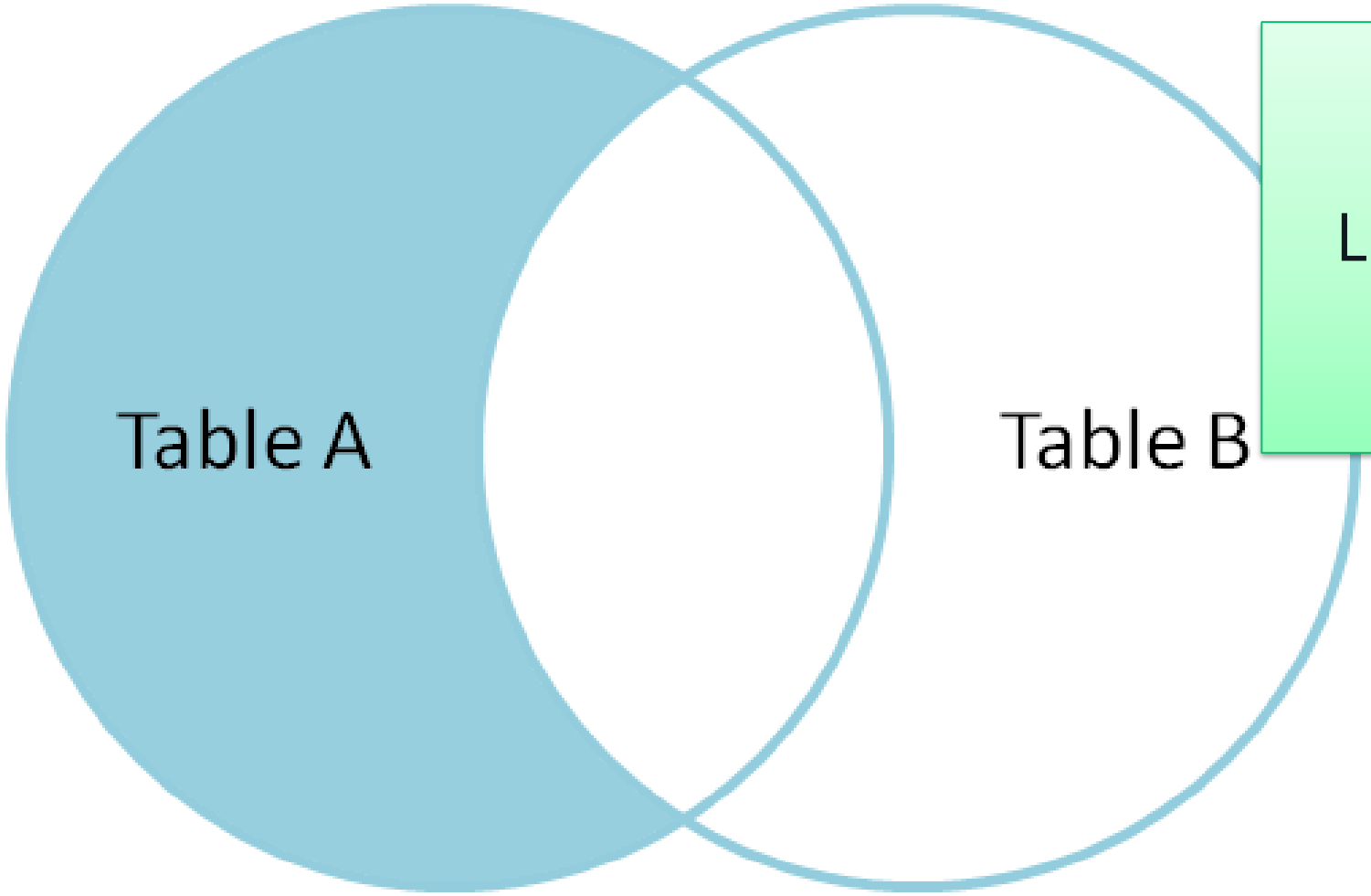

```
SELECT *
    FROM TableA AS A
FULL OUTER JOIN TableB AS B
        ON A.ID = B.ID
```

Joins – Left Outer Join



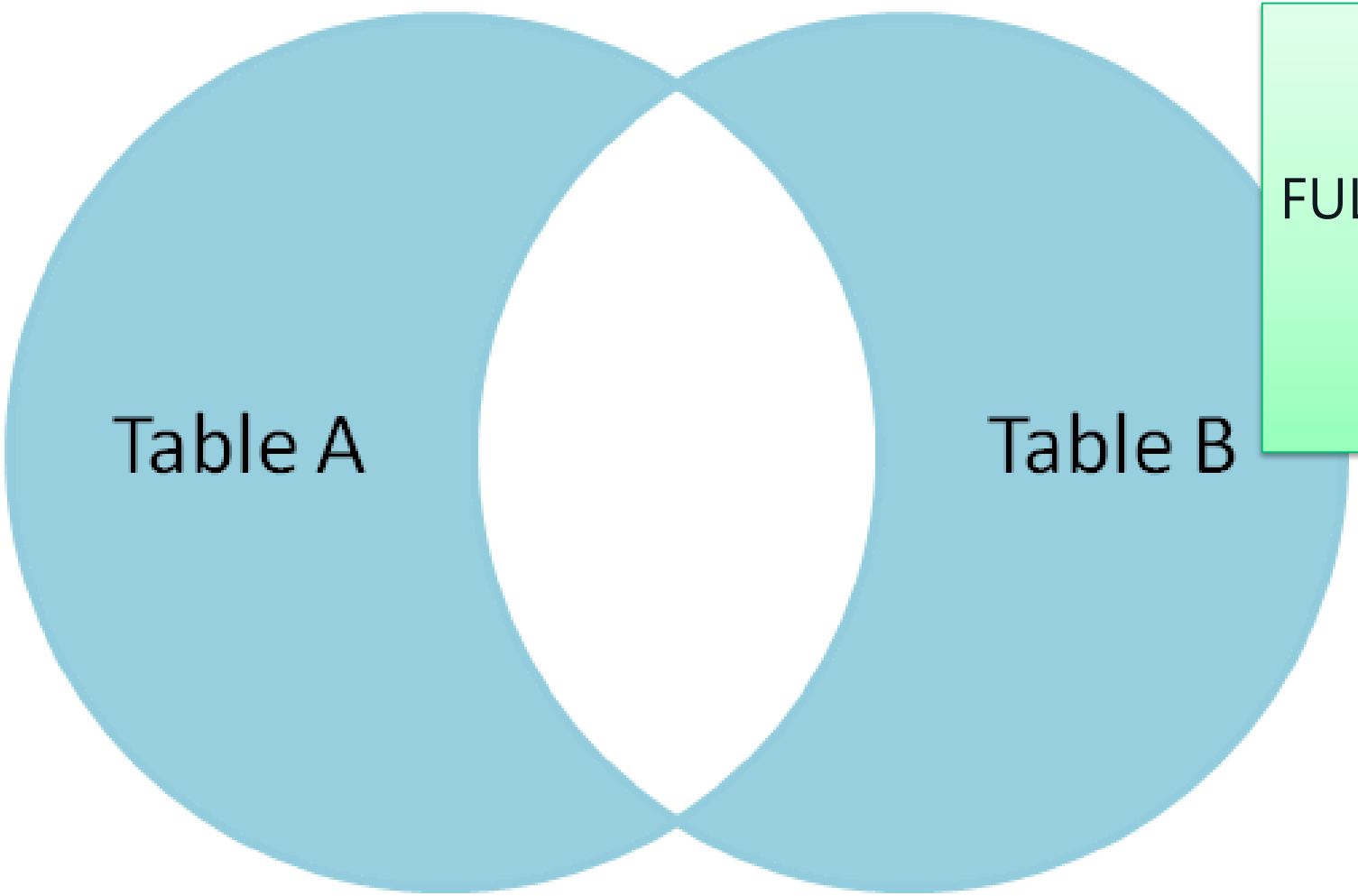

```
SELECT *
    FROM TableA AS A
LEFT JOIN TableB AS B
        ON A.ID = B.ID
```

Joins – Left Outer Join



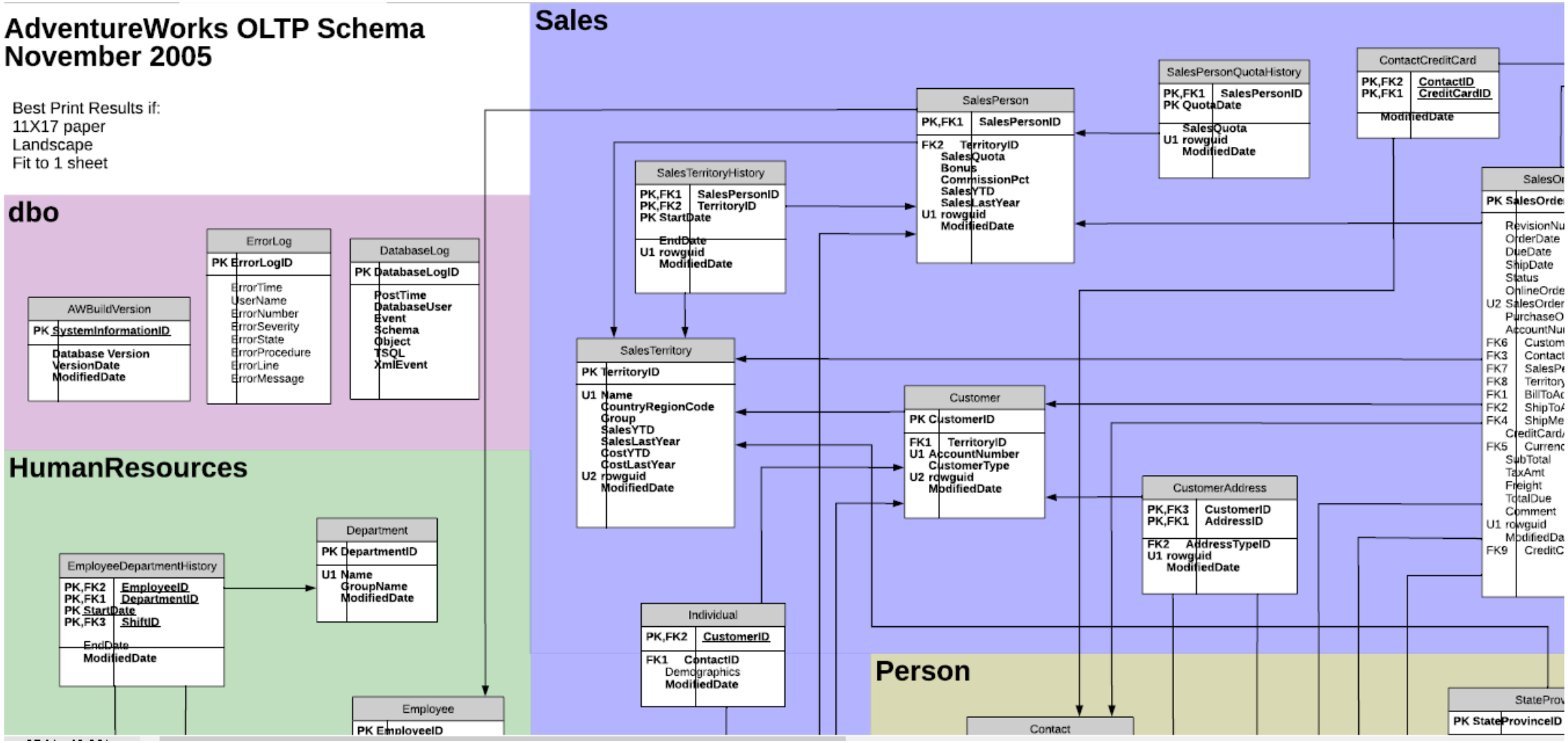

```
SELECT *
    FROM TableA AS A
LEFT JOIN TableB AS B
        ON A.ID = B.ID
WHERE B.ID IS NULL
```

Joins – Full Outer Join




```
SELECT *
FROM TableA AS A
FULL OUTER JOIN TableB AS B
ON A.ID = B.ID
WHERE A.ID IS NULL
OR B.ID IS NULL
```

DEMO


AdventureWorks OLTP Schema
November 2005
Best Print Results if:
11X17 paper
Landscape
Fit to 1 sheet
dbo
ErrorLog
PK ErrorLogID
ErrorTime
UserName
ErrorNumber
ErrorSeverity
ErrorState
ErrorProcedure
ErrorLine
ErrorMessage
DatabaseLog
PK DatabaseLogID
PostTime
DatabaseUser
Event
Schema
Object
TSQL
XmlEvent
AWBuildVersion
PK SystemInformationID
Database Version
VersionDate
ModifiedDate
HumanResources
Department
PK DepartmentID
U1 Name
GroupName
ModifiedDate
EmployeeDepartmentHistory
PK,FK2 EmployeeID
PK,FK1 DepartmentID
PK StartDate
PK,FK3 ShiftID
EndDate
ModifiedDate
Employee
PK EmployeeID
Sales
SalesPerson
PK,FK1 SalesPersonID
FK2 TerritoryID
SalesQuota
Bonus
CommissionPct
SalesYTD
SalesLastYear
U1 rowguid
ModifiedDate
SalesPersonQuotaHistory
PK,FK1 SalesPersonID
PK QuotaDate
SalesQuota
U1 rowguid
ModifiedDate
ContactCreditCard
PK,FK2 ContactID
PK,FK1 CreditCardID
ModifiedDate
SalesTerritoryHistory
PK,FK1 SalesPersonID
PK,FK2 TerritoryID
PK StartDate
EndDate
U1 rowguid
ModifiedDate
SalesTerritory
PK TerritoryID
U1 Name
CountryRegionCode
Group
SalesYTD
SalesLastYear
CostYTD
CostLastYear
U2 rowguid
ModifiedDate
Customer
PK CustomerID
FK1 TerritoryID
U1 AccountNumber
CustomerType
U2 rowguid
ModifiedDate
CustomerAddress
PK,FK3 CustomerID
PK,FK1 AddressID
FK2 AddressTypeID
U1 rowguid
ModifiedDate
Individual
PK,FK2 CustomerID
FK1 ContactID
Demographics
ModifiedDate
Person
Contact

References

The Language of SQL by Larry Rockoff

ISBN-13: 978-1-4354-5751-5

Querying Data with Transact-SQL by Itzik Ben-Gan

ISBN-13: 978-1-5093-0433-2

A Visual Explanation of SQL Joins by Jeff Atwood

http://blog.codinghorror.com/a-visual-explanation-of-sql-joins/

Links for Demo

SQL Operations Studio: https://docs.microsoft.com/en-us/sql/sql-operations-studio/download

SQL Server 2017 Express LocalDB: https://download.microsoft.com/download/E/F/2/EF23C21D-7860-4F05-88CE-39AA114B014B/SqlLocalDB.msi

AdventureWorks OLTP Database Backup: https://github.com/Microsoft/sql-server-samples/releases/download/adventureworks/AdventureWorks2017.bak

AdventureWorks OLTP Database Diagram: https://www.microsoft.com/en-us/download/details.aspx?id=10331