SQL
Session 3

Table of Contents

JOINs

Important Concepts

Primary Key (PK):

The primary key is a column in our table that makes each row (aka, record) unique.

Foreign Key (FK):

Foreign key is a column in a table that uniquely identifies each row of another table. That column refers to a primary key of another table. This creates a kind of link between the tables.

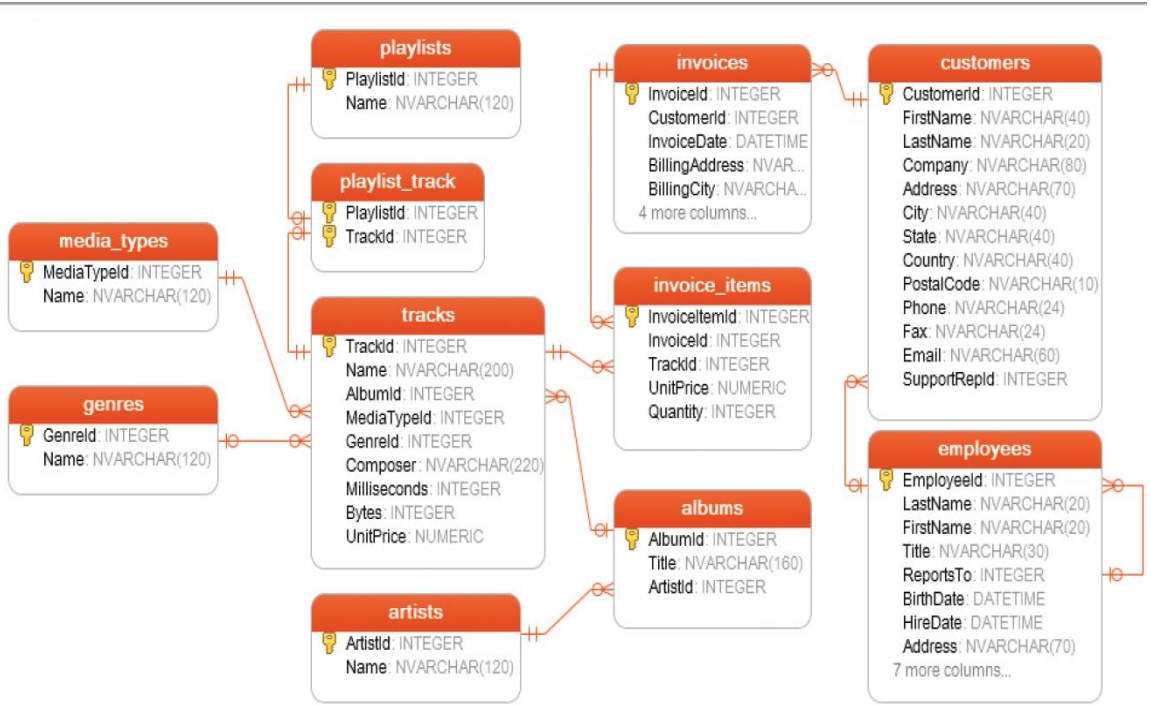


Table of Contents

1 Introduction

Introduction

A JOIN clause is used to combine two or more tables into a single table.

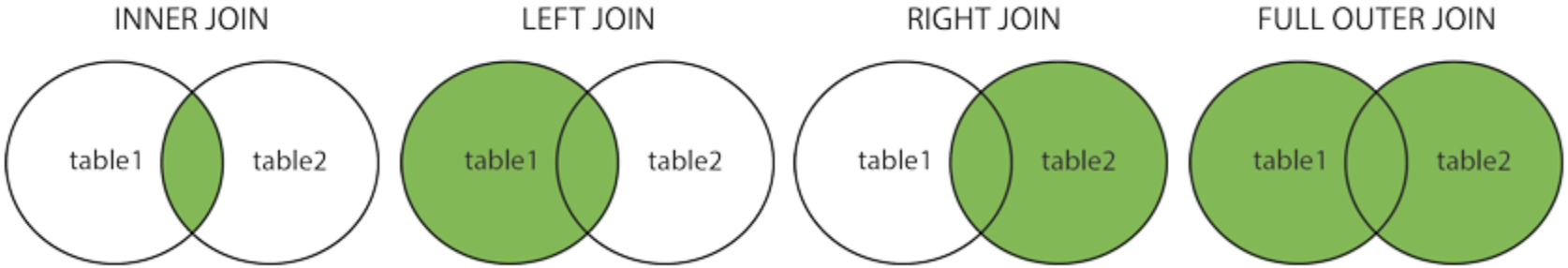
Joins are usually applied based on the keys that define the relationship between those tables or on common fields.

Introduction

In most cases this joins are created using the primary key of one table and the foreign key of the other table we want to join it with.

2 JOIN Types

JOIN Types

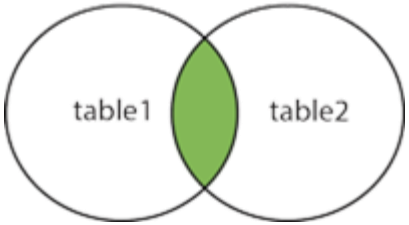


- **INNER JOIN:** Returns the common records in both tables.
- **LEFT OUTER JOIN:** Returns all records from the left table and matching records from the right table.
- **RIGHT OUTER JOIN:** Returns all records from the right table and matching records from the left table.
- **FULL OUTER JOIN:** Returns all records of both left and right tables.

3 INNER JOIN

INNER JOIN

INNER JOIN is the most common type of JOINs. The INNER JOIN selects records that have matching values in both tables. INNER keyword is optional for this type of JOIN.



Syntax

```
1  SELECT columns
2    FROM table_A
3    INNER JOIN table_B ON join_conditions
```

join_conditions

table_A. common_field = table_B.common_field

students

name	exam	score
John	SQL	75
Mary	AWS	80
Clark	Python	60

tests

exam	passing_score
SQL	70
AWS	80
Python	70
Network	60

```
SELECT students.name, students.exam,
       students.score, tests.passing_score
FROM students
INNER JOIN tests ON students.exam = tests.exam;
```

students

name	exam	score
John	SQL	75
Mary	AWS	80
Clark	Python	60

tests

exam	passing_score
SQL	70
AWS	80
Python	70
Network	60

```
SELECT students.name, students.exam,
       students.score, tests.passing_score
FROM students
INNER JOIN tests ON students.exam = tests.exam;
```

students

name	exam	score
John	SQL	75
Mary	AWS	80
Clark	Python	60

tests

exam	passing_score
SQL	70
AWS	80
Python	70
Network	60

```
SELECT students.name, students.exam,
       students.score, tests.passing_score
FROM students
INNER JOIN tests ON students.exam = tests.exam;
```

students

name	exam	score
John	SQL	75
Mary	AWS	80
Clark	Python	60

tests

exam	passing_score
SQL	70
AWS	80
Python	70
Network	60

```
SELECT students.name, students.exam,
       students.score, tests.passing_score
FROM students
INNER JOIN tests ON students.exam = tests.exam;
```

students **tests**

name	exam	score	exam	passing_score
John	SQL	75	SQL	70
Mary	AWS	80	AWS	80
Clark	Python	60	Python	70

```
SELECT students.name, students.exam,
       students.score, tests.passing_score
FROM students
INNER JOIN tests ON students.exam = tests.exam;
```

output of the query

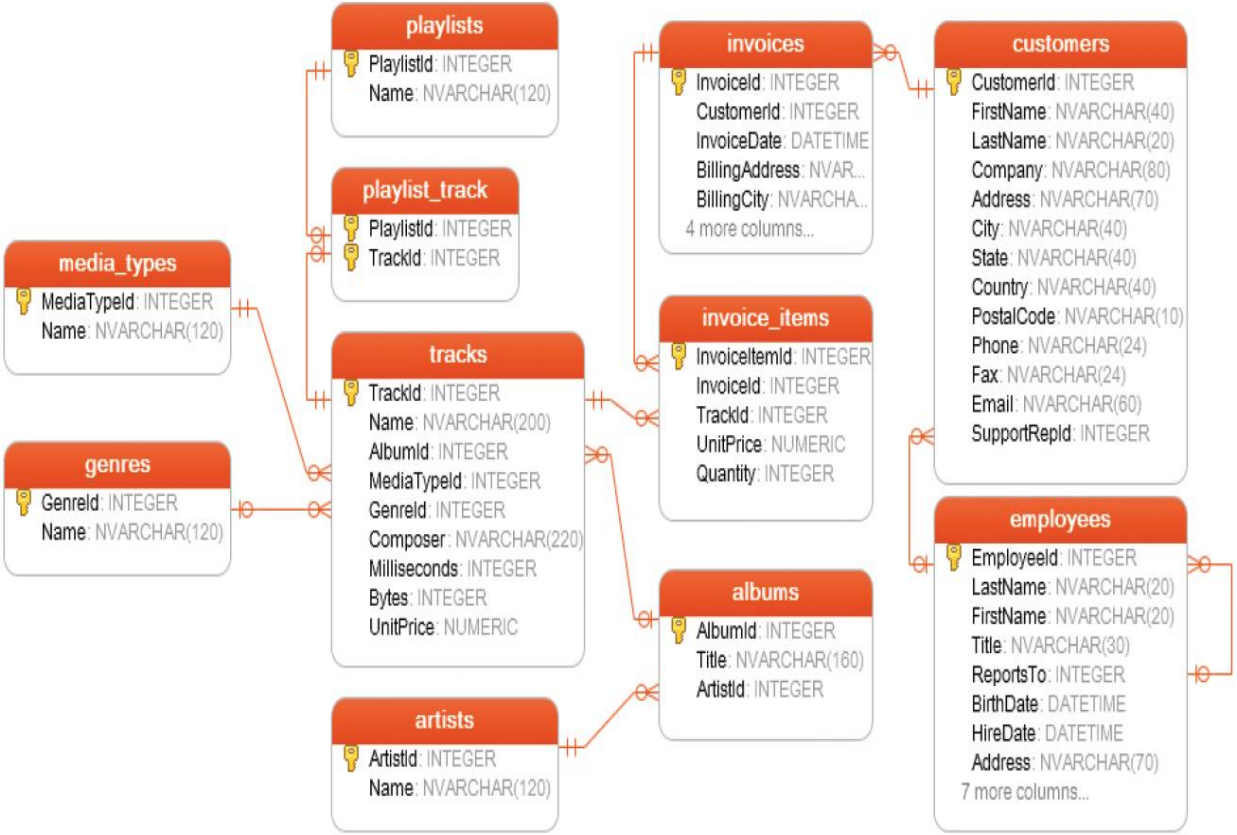
name	exam	score	passing_score
John	SQL	75	70
Mary	AWS	80	80
Clark	Python	60	70

INNER JOIN

Syntax of Join of Multiple Tables

```
SELECT columns
  FROM table_A
  INNER JOIN table_B
    ON join_conditions1 AND join_conditions2
  INNER JOIN table_C
    ON join_conditions3 OR join_conditions4
...
```

Find the genre of each track.



tracks

	TrackId	Name	AlbumId	MediaTypeId	GenreId
1	1	For Those About To Rock (We Salute You)	1	1	1
2	2	Balls to the Wall	2	2	1
3	3	Fast As a Shark	3	2	1
4	4	Restless and Wild	3	2	1
5	5	Princess of the Dawn	3	2	1
6	6	Put The Finger On You	1	1	1
7	7	Let's Get It Up	1	1	1
8	8	Inject The Venom	1	1	1
9	9	Snowballed	1	1	1
10	10	Evil Walks	1	1	1

genres

	GenreId	Name
1	1	Rock
2	2	Jazz
3	3	Metal
4	4	Alternative & Punk
5	5	Rock And Roll
6	6	Blues
7	7	Latin
8	8	Reggae
9	9	Pop
10	10	Soundtrack

tracks

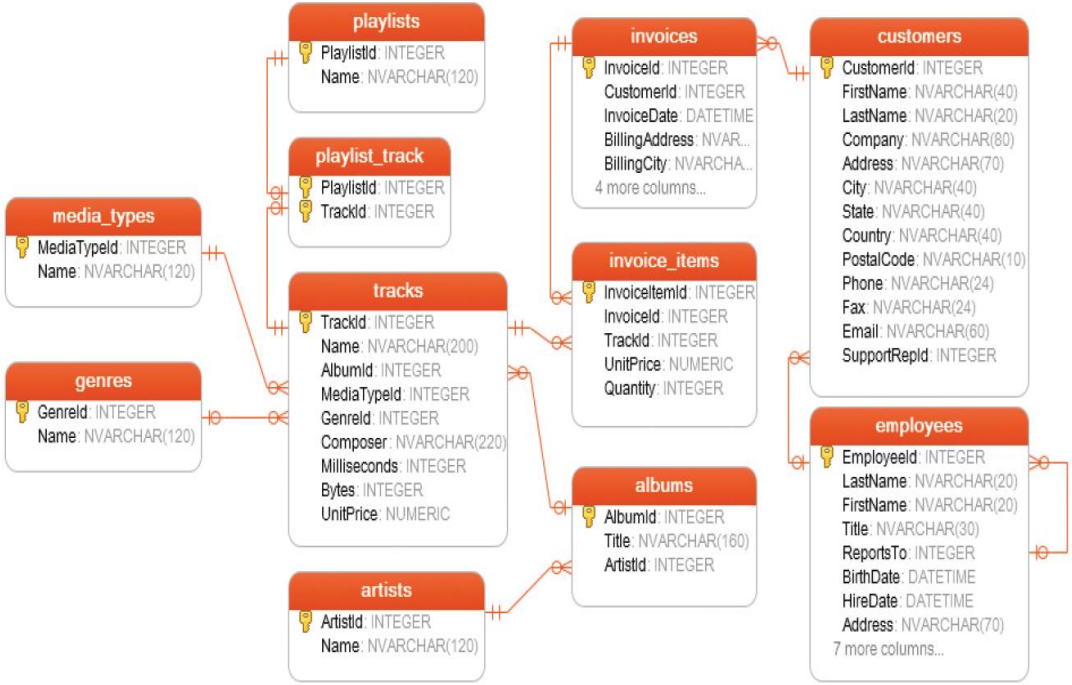
	TrackId	Name	AlbumId	MediaTypeId	GenreId
1	1	For Those About To Rock (We Salute You)	1	1	1
2	2	Balls to the Wall	2	2	1
3	3	Fast As a Shark	3	2	1
4	4	Restless and Wild	3	2	1
5	5	Princess of the Dawn	3	2	1
6	6	Put The Finger On You	1	1	1
7	7	Let's Get It Up	1	1	1
8	8	Inject The Venom	1	1	1
9	9	Snowballed	1	1	1
10	10	Evil Walks	1	1	1

genres

	GenreId	Name
1	1	Rock
2	2	Jazz
3	3	Metal
4	4	Alternative & Punk
5	5	Rock And Roll
6	6	Blues
7	7	Latin
8	8	Reggae
9	9	Pop
10	10	Soundtrack

	Name	Name
1	For Those About To Rock (We Salute You)	Rock
2	Balls to the Wall	Rock
3	Fast As a Shark	Rock
4	Restless and Wild	Rock
5	Princess of the Dawn	Rock
6	Put The Finger On You	Rock
7	Let's Get It Up	Rock
8	Inject The Venom	Rock
9	Snowballed	Rock
10	Evil Walks	Rock

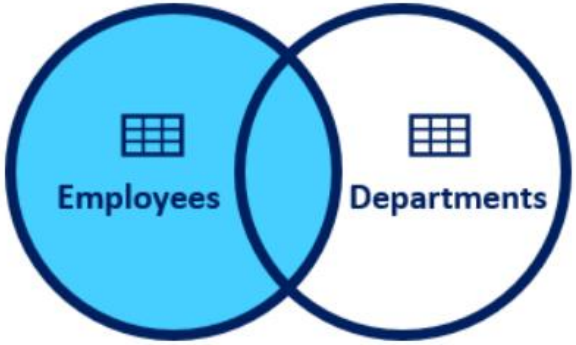
Find the customer name of each invoice.
Your result will include Invoice id and customer name.



LEFT JOIN

LEFT JOIN

In this JOIN statement, all the records of the left table and the common records of the right table are returned in the query. If no matching rows are found in the right table during the JOIN operation, these values are assigned as NULL.



Visual Representation of Left JOIN

Syntax

```
SELECT columns
  FROM table_A
  LEFT JOIN table_B ON join_conditions
```

join_conditions

table_A. common_field = table_B.common_field

students

name	exam	score
John	SQL	75
Mary	AWS	80
Clark	Python	60

tests

exam	passing_score
SQL	70
AWS	80
Python	70
Network	60

```
SELECT tests.exam, tests.passing_score,
       students.name, students.score
FROM tests
LEFT JOIN students ON tests.exam = students.exam;
```

tests

exam	passing_score
SQL	70
AWS	80
Python	70
Network	60

students

name	exam	score
John	SQL	75
Mary	AWS	80
Clark	Python	60

```
SELECT tests.exam, tests.passing_score,
       students.name, students.score
FROM tests
LEFT JOIN students ON tests.exam = students.exam;
```

tests

exam	passing_score
SQL	70
AWS	80
Python	70
Network	60

students

name	exam	score
John	SQL	75
Mary	AWS	80
Clark	Python	60

```
SELECT tests.exam, tests.passing_score,
       students.name, students.score
FROM tests
LEFT JOIN students ON tests.exam = students.exam;
```

tests **students**

exam	passing_score	name	exam	score
SQL	70	John	SQL	75
AWS	80	Mary	AWS	80
Python	70	Clark	Python	60
Network	60	Null	Null	Null

```
SELECT tests.exam, tests.passing_score,
       students.name, students.score
FROM tests
LEFT JOIN students ON tests.exam = students.exam;
```

tests **students**

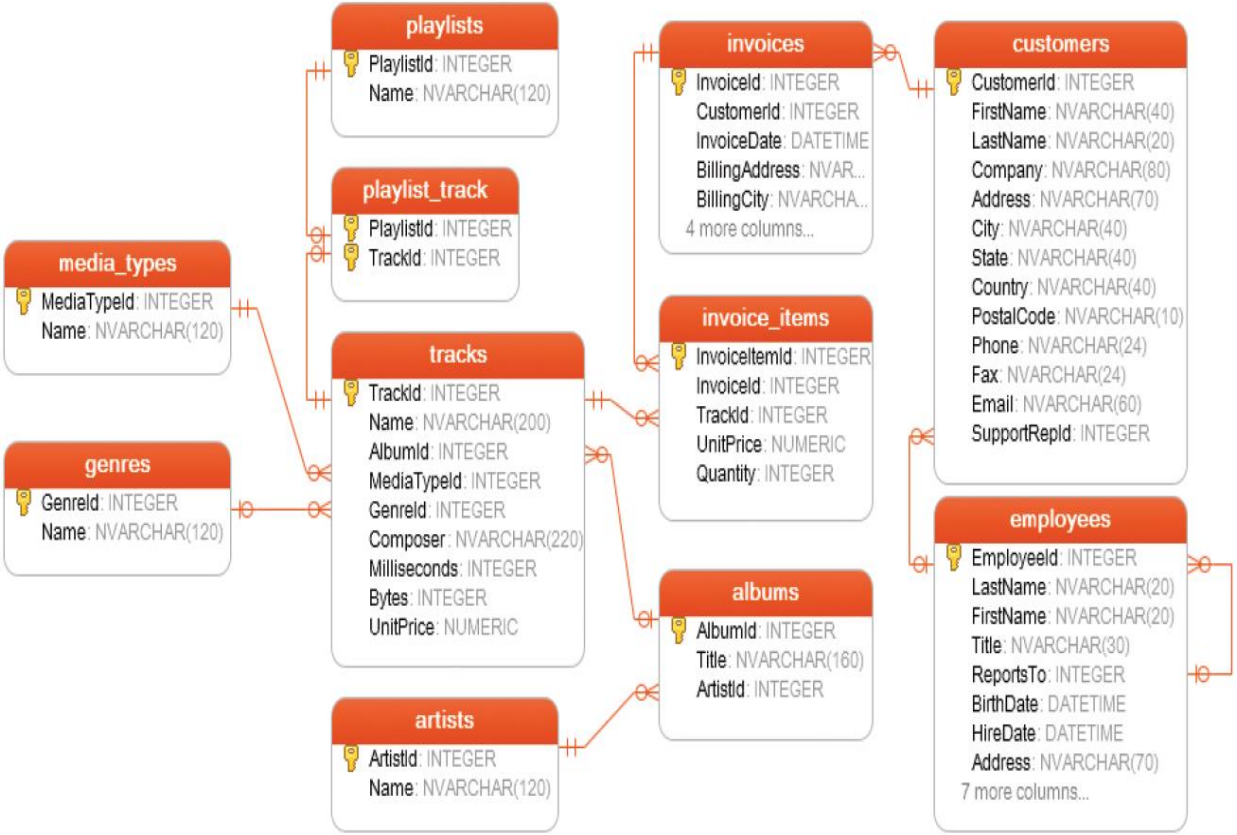
exam	passing_score	name	exam	score
SQL	70	John	SQL	75
AWS	80	Mary	AWS	80
Python	70	Clark	Python	60
Network	60	Null	Null	Null

```
SELECT tests.exam, tests.passing_score,
       students.name, students.score
FROM tests
LEFT JOIN students ON tests.exam = students.exam;
```

output of the query

exam	passing_score	name	score
SQL	70	John	75
AWS	80	Mary	80
Python	70	Clark	60
Network	60	Null	Null

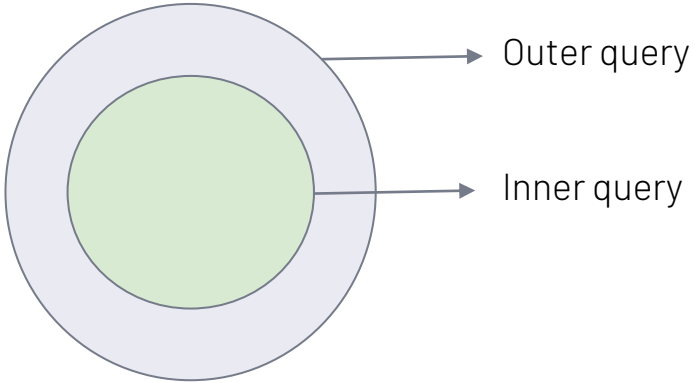
Find the artists' album info



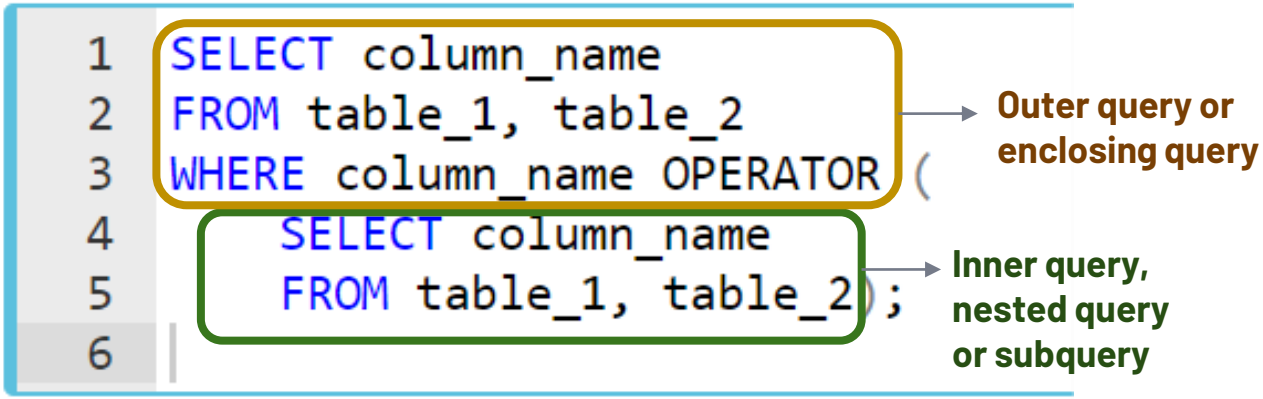
Subqueries

Introduction

A subquery is a SELECT statement that is nested within another statement. The subquery is also called the inner query or nested query.



Syntax



- Subqueries are nested queries that provide data to the enclosing query.
- Subqueries can return individual values or a list of records
- Subqueries must be enclosed with parentheses

Types of Subqueries

There are two main types of subqueries:

- Single-row subqueries
- Multiple-row subqueries

Single-row Subqueries

Single-row subqueries return one row with only one column and are typically used with single-row operators such as =, >, >=, <=, <>, != especially in WHERE clause.

Example

Find the employees who get paid more than Rodney Weaver

employees table

	emp_id	first_name	last_name	salary	job_title	gender	hire_date
1	17679	Robert	Gilmore	110000	Operations Director	Male	2018-09-04
2	26650	Elvis	Ritter	86000	Sales Manager	Male	2017-11-24
3	30840	David	Barrow	85000	Data Scientist	Male	2019-12-02
4	49714	Hugo	Forester	55000	IT Support Specialist	Male	2019-11-22
5	51821	Linda	Foster	95000	Data Scientist	Female	2019-04-29
6	67323	Lisa	Wiener	75000	Business Analyst	Female	2018-08-09
7	70950	Rodney	Weaver	87000	Project Manager	Male	2018-12-20
8	71329	Gayle	Meyer	77000	HR Manager	Female	2019-06-28
9	76589	Jason	Christian	99000	Project Manager	Male	2019-01-21
10	97927	Billie	Lanning	67000	Web Developer	Female	2018-06-25

query:

```
1  SELECT first_name, last_name, salary
2  FROM employees
3  WHERE salary >
4      (SELECT salary
5       FROM employees
6       WHERE first_name = "Rodney");
7
```

output:

```
1  first_name  last_name   salary
2  ----------  ----------  ----------
3  Robert      Gilmore     110000
4  Linda       Foster      95000
5  Jason       Christian   99000
```

Analyze the query-1

```
1  SELECT first_name, last_name, salary
2  FROM employees
3  WHERE salary >
4      (SELECT salary
5       FROM employees
6       WHERE first_name = "Rodney");
7
```

1

employees table

	emp_id	first_name	last_name	salary	job_title	gender	hire_date
1	17679	Robert	Gilmore	110000	Operations Director	Male	2018-09-04
2	26650	Elvis	Ritter	86000	Sales Manager	Male	2017-11-24
3	30840	David	Barrow	85000	Data Scientist	Male	2019-12-02
4	49714	Hugo	Forester	55000	IT Support Specialist	Male	2019-11-22
5	51821	Linda	Foster	95000	Data Scientist	Female	2019-04-29
6	67323	Lisa	Wiener	75000	Business Analyst	Female	2018-08-09
7	[illegible]	Rodney	Weaver	87000	Project Manager	Male	2018-12-20
8	71329	Gayle	Meyer	77000	HR Manager	Female	2019-06-28
9	76589	Jason	Christian	99000	Project Manager	Male	2019-01-21
10	97927	Billie	Lanning	67000	Web Developer	Female	2018-06-25

1 The inner query is executed first and returns 87000 which is the salary of Rodney.

Analyze the query-2

```
2  SELECT first_name, last_name, salary
   FROM employees
   WHERE salary >
       (SELECT salary
        FROM employees
        WHERE first_name = "Rodney");
1
```

employees table

	emp_id	first_name	last_name	salary	job_title	gender	hire_date
1	17679	Robert	Gilmore	110000	Operations Director	Male	2018-09-04
2	26650	Elvis	Ritter	86000	Sales Manager	Male	2017-11-24
3	30840	David	Barrow	85000	Data Scientist	Male	2019-12-02
4	49714	Hugo	Forester	55000	IT Support Specialist	Male	2019-11-22
5	51821	Linda	Foster	95000	Data Scientist	Female	2019-04-29
6	67323	Lisa	Wiener	75000	Business Analyst	Female	2018-08-09
7	70938	Rodney	Weaver	87000	Project Manager	Male	2018-12-20
8	71329	Gayle	Meyer	77000	HR Manager	Female	2019-06-28
9	76589	Jason	Christian	99000	Project Manager	Male	2019-01-21
10	97927	Billie	Lanning	67000	Web Developer	Female	2018-06-25

1. The inner query is execute first and returns 87000 which is the salary of Rodney.

2. The value 87000 is passed to the outer query, in particular to the WHERE clause.

Analyze the query-3

```
2  SELECT first_name, last_name, salary
   FROM employees
   WHERE salary > 87000
   (SELECT salary
1   FROM employees
    WHERE first_name = "Rodney");
```

employees table

	emp_id	first_name	last_name	salary	job_title	gender	hire_date
1	17679	Robert	Gilmore	110000	Operations Director	Male	2018-09-04
2	26650	Elvis	Ritter	86000	Sales Manager	Male	2017-11-24
3	30840	David	Barrow	85000	Data Scientist	Male	2019-12-02
4	49714	Hugo	Forester	55000	IT Support Specialist	Male	2019-11-22
5	51821	Linda	Foster	95000	Data Scientist	Female	2019-04-29
6	67323	Lisa	Wiener	75000	Business Analyst	Female	2018-08-09
7	70938	Rodney	Weaver	87000	Project Manager	Male	2018-12-20
8	71329	Gayle	Meyer	77000	HR Manager	Female	2019-06-28
9	76589	Jason	Christian	99000	Project Manager	Male	2019-01-21
10	97927	Billie	Lanning	67000	Web Developer	Female	2018-06-25

1. The inner query is execute first and returns 87000 which is the salary of Rodney.

2. The value 87000 is passed to the outer query, in particular to the WHERE clause.

Analyze the query-4

```sql
SELECT first_name, last_name, salary
FROM employees
WHERE salary > 87000
    (SELECT salary
     FROM employees
     WHERE first_name = "Rodney");
```

output:

first_name	last_name	salary
Robert	Gilmore	110000
Linda	Foster	95000
Jason	Christian	99000

1. The inner query is execute first and returns 87000 which is the salary of Rodney.

2. The value 87000 is passed this value to the outer query, in particular to the WHERE clause.

Example

Find out the employees who get paid more than the average salary

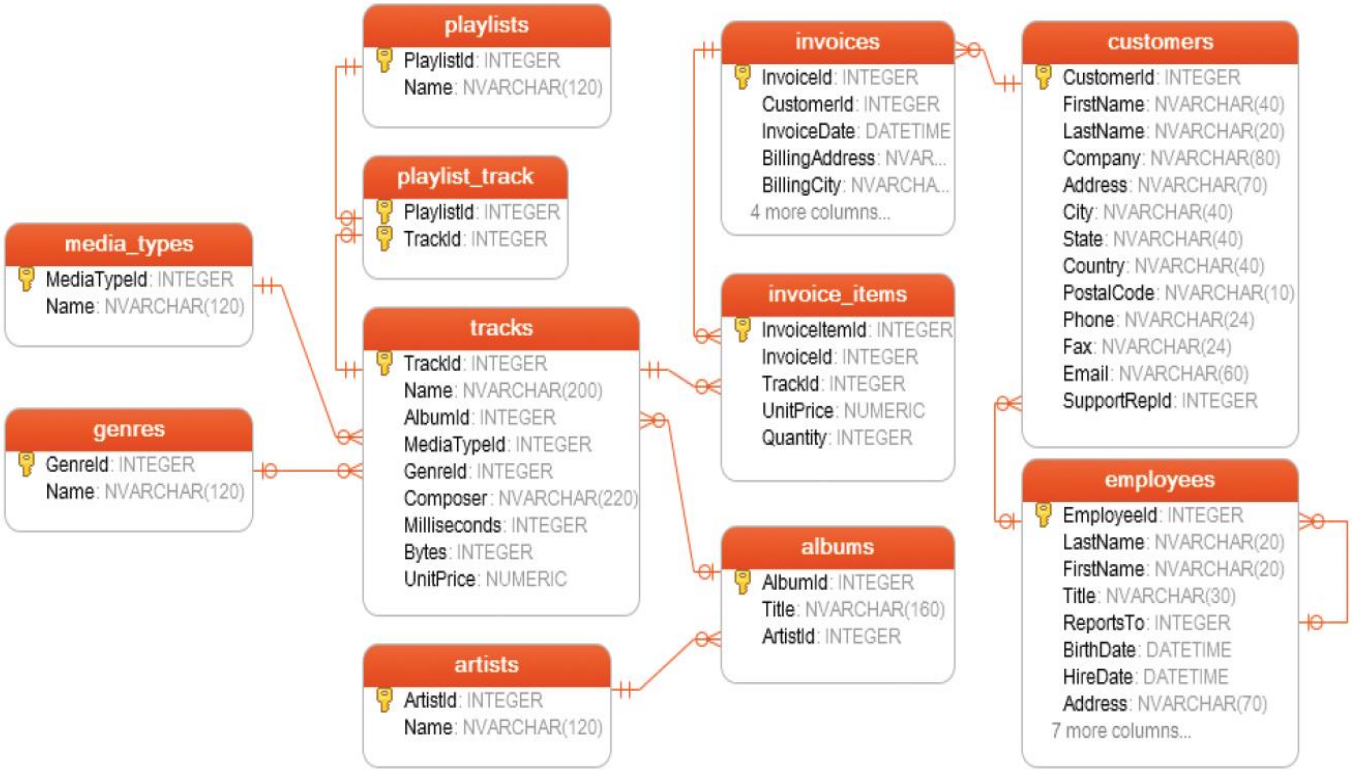
employees table

	emp_id	first_name	last_name	salary	job_title	gender	hire_date
1	17679	Robert	Gilmore	110000	Operations Director	Male	2018-09-04
2	26650	Elvis	Ritter	86000	Sales Manager	Male	2017-11-24
3	30840	David	Barrow	85000	Data Scientist	Male	2019-12-02
4	49714	Hugo	Forester	55000	IT Support Specialist	Male	2019-11-22
5	51821	Linda	Foster	95000	Data Scientist	Female	2019-04-29
6	67323	Lisa	Wiener	75000	Business Analyst	Female	2018-08-09
7	70950	Rodney	Weaver	87000	Project Manager	Male	2018-12-20
8	71329	Gayle	Meyer	77000	HR Manager	Female	2019-06-28
9	76589	Jason	Christian	99000	Project Manager	Male	2019-01-21
10	97927	Billie	Lanning	67000	Web Developer	Female	2018-06-25

```
1  SELECT first_name, last_name, salary
2  FROM employees
3  WHERE salary >
4      (SELECT AVG(salary)
5       FROM employees);
```

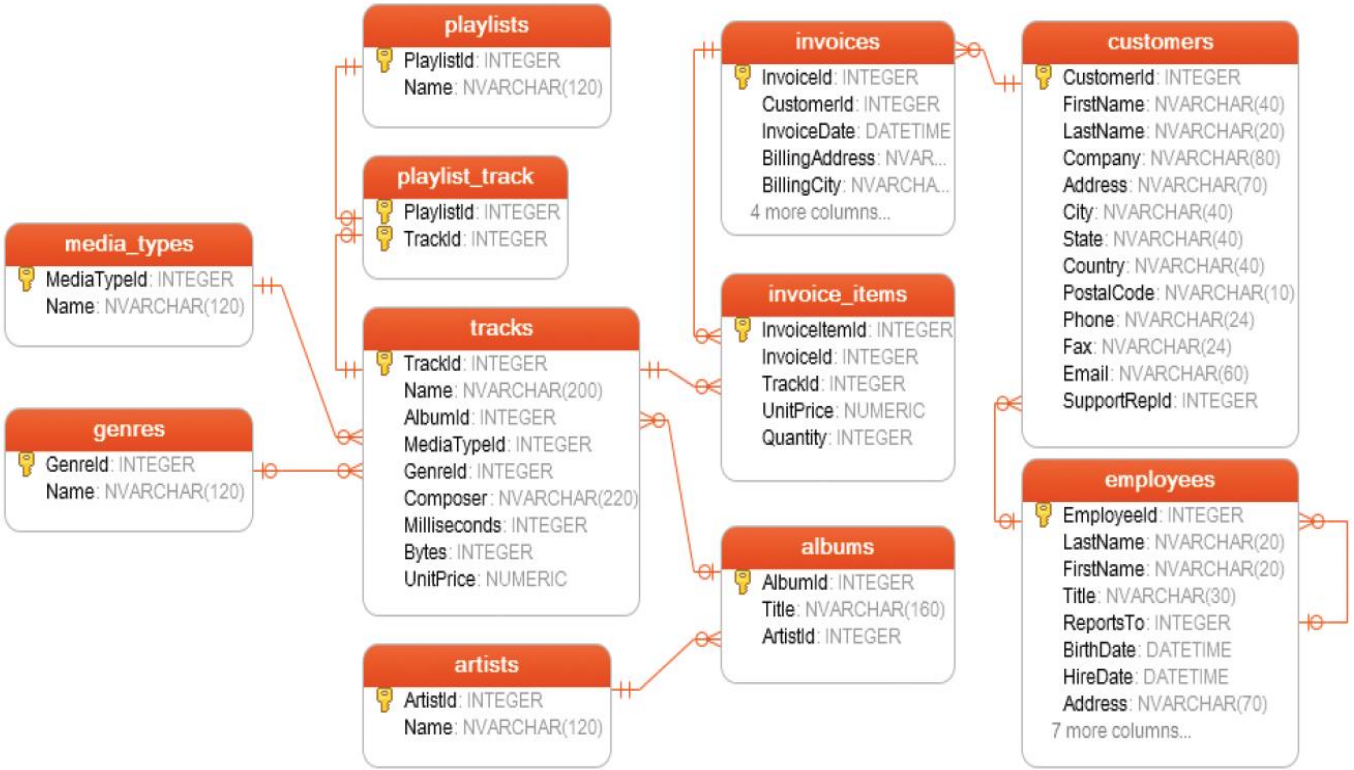
Query Time

Retrieve track id, track name, album id info of the Album title 'Faceless'. (**use : Subquery**)



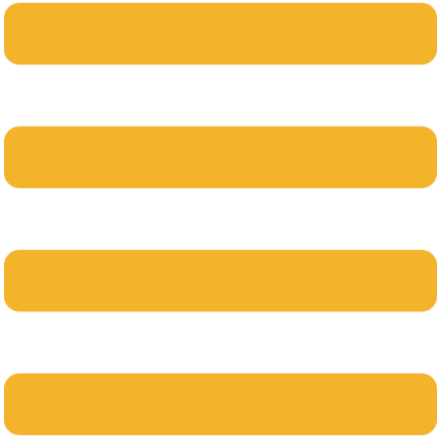
Most queries using a join can be rewritten using a subquery (a query nested within another query), and most subqueries can be rewritten as joins.

Retrieve track id, track name, album id info of the Album title 'Faceless'. (**use : Joins**)



Multiple-row Subqueries

Multiple-row subqueries return sets of rows and are used with multiple-row operators such as **`IN`**, **`NOT IN`**, **`ANY`**, **`ALL`**.



Example

Find the employees (first name, last name from employees table) who work under the Operations department (departments table)

employees table

	emp_id	first_name	last_name	salary	job_title	gender	hire_date
1	17679	Robert	Gilmore	110000	Operations Director	Male	2018-09-04
2	26650	Elvis	Ritter	86000	Sales Manager	Male	2017-11-24
3	30840	David	Barrow	85000	Data Scientist	Male	2019-12-02
4	49714	Hugo	Forester	55000	IT Support Specialist	Male	2019-11-22
5	51821	Linda	Foster	95000	Data Scientist	Female	2019-04-29
6	67323	Lisa	Wiener	75000	Business Analyst	Female	2018-08-09
7	70950	Rodney	Weaver	87000	Project Manager	Male	2018-12-20
8	71329	Gayle	Meyer	77000	HR Manager	Female	2019-06-28
9	76589	Jason	Christian	99000	Project Manager	Male	2019-01-21
10	97927	Billie	Lanning	67000	Web Developer	Female	2018-06-25

departments table

	emp_id	dept_name	dept_id
1	17679	Operations	13
2	26650	Marketing	14
3	30840	Operations	13
4	49823	Technology	12
5	51821	Operations	13
6	67323	Marketing	14
7	71119	Administrative	11
8	76589	Operations	13
9	97927	Technology	12

query:

```sql
SELECT first_name, last_name
FROM employees
WHERE emp_id IN
   (SELECT emp_id
    FROM departments
    WHERE dept_name = 'Operations');
```

output:

```
first_name  last_name
----------  ----------
Robert      Gilmore
David       Barrow
Linda       Foster
Jason       Christian
```

Analyze the query-1

```sql
SELECT first_name, last_name
FROM employees
WHERE emp_id IN
    (SELECT emp_id
     FROM departments
     WHERE dept_name = 'Operations');
```

1

departments table

	emp_id	dept_name	dept_id
1	17679	Operations	13
2	26650	Marketing	14
3	30840	Operations	13
4	49823	Technology	12
5	51821	Operations	13
6	67323	Marketing	14
7	71119	Administrative	11
8	76589	Operations	13
9	97927	Technology	12

1 The inner query returns the employees ids who work under the Operations department

Analyze the query-2

```
2  SELECT first_name, last_name
   FROM employees
   WHERE emp_id IN
4    (SELECT emp_id
1     FROM departments
      WHERE dept_name = 'Operations');
7
```

departments table

	emp_id	dept_name	dept_id
1	17679	Operations	13
2	26650	Marketing	14
3	30840	Operations	13
4	49823	Technology	12
5	51821	Operations	13
6	67323	Marketing	14
7	71119	Administrative	11
8	76589	Operations	13
9	97927	Technology	12

1. The inner query returns the employees ids who work under the Operations department
2. Employees ids are passed to the outer query.

Analyze the query-3

```
SELECT first_name, last_name
FROM employees
WHERE emp_id IN (17679, 30840, 51821, 76589)
    (SELECT emp_id
     FROM departments
     WHERE dept_name = 'Operations');
```

departments table

	emp_id	dept_name	dept_id
1	17679	Operations	13
2	26650	Marketing	14
3	30840	Operations	13
4	49823	Technology	12
5	51821	Operations	13
6	67323	Marketing	14
7	71119	Administrative	11
8	76589	Operations	13
9	97927	Technology	12

1. The inner query returns the employees ids who work under the Operations department
2. Employees ids are passed to the outer query.

Analyze the query-4

```
1  SELECT first_name, last_name
2  FROM employees
3  WHERE emp_id IN (17679, 30840, 51821, 76589)
4     (SELECT emp_id
5      FROM departments
6      WHERE dept_name = 'Operations');
7
```

employees table

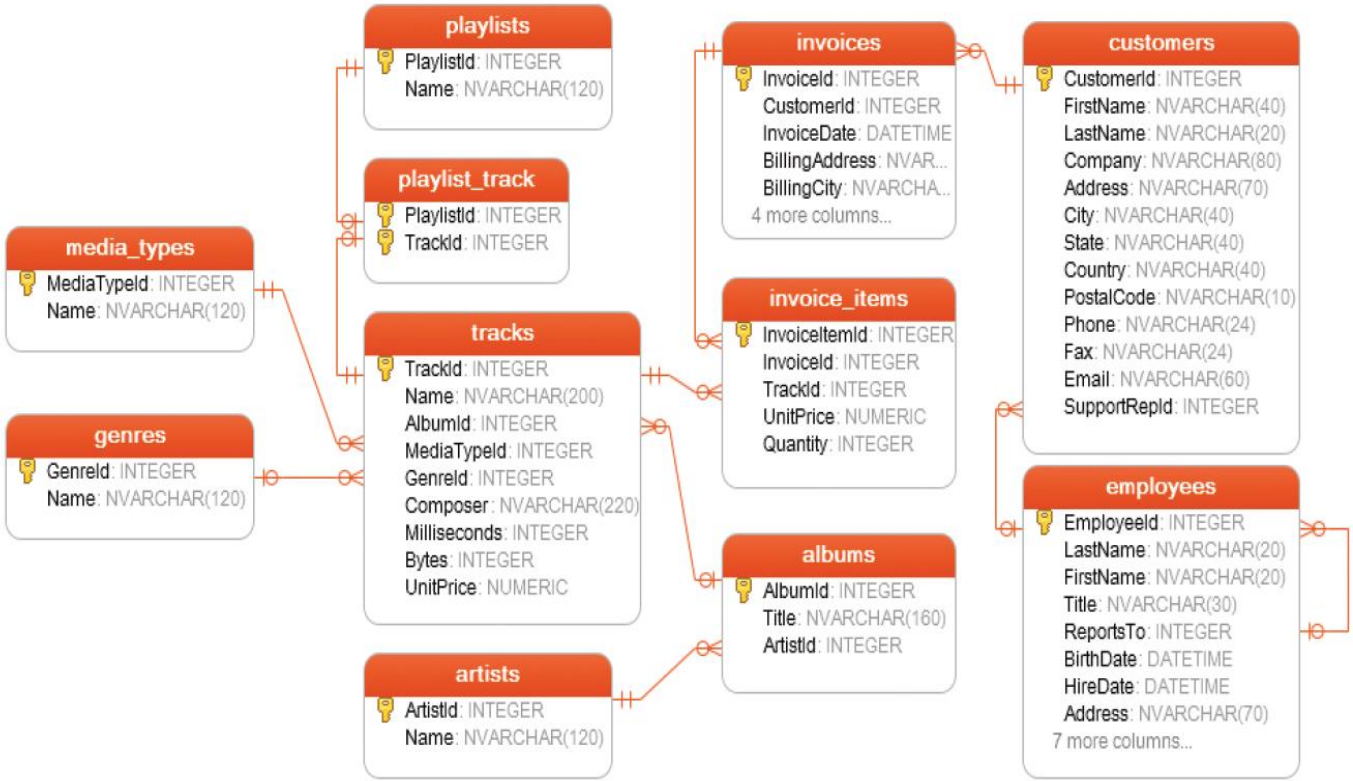
	emp_id	first_name	last_name	salary	job_title	gender	hire_date
1	17679	Robert	Gilmore	110000	Operations Director	Male	2018-09-04
2	26650	Elvis	Ritter	86000	Sales Manager	Male	2017-11-24
3	30840	David	Barrow	85000	Data Scientist	Male	2019-12-02
4	49714	Hugo	Forester	55000	IT Support Specialist	Male	2019-11-22
5	51821	Linda	Foster	95000	Data Scientist	Female	2019-04-29
6	67323	Lisa	Wiener	75000	Business Analyst	Female	2018-08-09
7	70950	Rodney	Weaver	87000	Project Manager	Male	2018-12-20
8	71329	Gayle	Meyer	77000	HR Manager	Female	2019-06-28
9	76589	Jason	Christian	99000	Project Manager	Male	2019-01-21
10	97927	Billie	Lanning	67000	Web Developer	Female	2018-06-25

output:

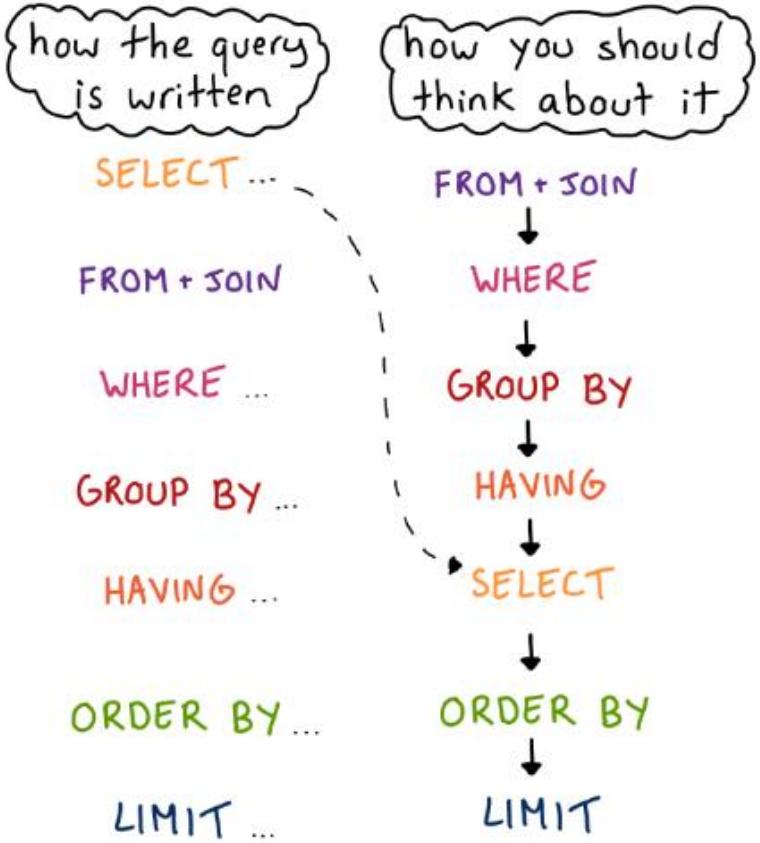
```
1  first_name  last_name
2  ----------  ----------
3  Robert      Gilmore
4  David       Barrow
5  Linda       Foster
6  Jason       Christian
7
```

Outer query filters those employees ids and returns their first name and last name as a result set.

Retrieve track id, track name, album id info of the Album title ‘Faceless’ and ‘Let There Be Rock’



The query's steps don't happen in the order they're written:



(In reality query execution is much more complicated than this. There are a lot of optimizations.)

THANKS!

Any questions?