

Chapter 9 Complex Joins

Outer Joins
Self Joins

Review from Ch 6

Inner Join (a.k.a. Equijoin)

- rows from one table are included in the result set only if there is a **matching** row in the other table
- rows won't be shown if there is no matching row in the other table
 - the **missing rows** phenomenon

Join Conditions

- indicate the common field that is used to join rows across the tables

SQL92

```
SQL> SELECT title, descr, length
      FROM article a, type t
      WHERE a.type = t.type
      ORDER BY descr
```

SQL99

```
SQL> SELECT title, descr, length
      FROM article a INNER JOIN type t
      ON a.type = t.type
      ORDER BY descr
```

But...which
DESCRs **don't**
you see?

2

Outer Join Concepts

- All rows from one table are included in the result set even if no matching row exists in the other table
 - rows that have a matching row in the other table are shown, as usual
 - when a row does not have a matching row in the other table, Oracle generates a Null-filled row to match it to
- Driving Table** (n1b)
 - the tables whose full set of records is selected
 - when all rows from Table A are to be included in the result set even if no matching rows exist in Table B, Table A is the **driving table**
- 2 ways to perform an Outer Join in Oracle
 - use ANSI99 **OUTER JOIN** in FROM clause
 - portable to non-Oracle databases
 - use the Oracle outer join operator **(+)** on the **non-driving table**
 - Oracle specific, the only available OJ method prior to Oracle version 9i

3

ANSI Outer Joins

SQL99

- Show article titles and their type descriptions, including types that don't have a matching article

SQL99

```
SQL> SELECT title, descr, length
      FROM type t LEFT OUTER JOIN article a
      ON t.type = a.type
      ORDER BY descr;
```

- Which rows have NULLs?
- Which columns have NULLs?

4

ANSI Outer Joins

SQL99

- Which is the driving table in each?

```
SQL> SELECT title, descr, length
      FROM type t LEFT OUTER JOIN article a
      ON t.type = a.type
      ORDER BY descr;
```

```
SQL> SELECT title, descr, length
      FROM article a RIGHT OUTER JOIN type t
      ON a.type = t.type
      ORDER BY descr;
```

```
SQL> SELECT title, descr, length
      FROM article a LEFT OUTER JOIN type t
      ON a.type = t.type
      ORDER BY descr;
```

5

ANSI Outer Joins

SQL99

- How can you show only those types that lack a matching article?

```
SQL> SELECT title, descr, length
      FROM type t LEFT OUTER JOIN article a
      ON t.type = a.type
      WHERE articlenum IS NULL
      ORDER BY descr;
```

- Add a WHERE clause that tests for IS NULL on a column from the non-driving table that cannot be null

6

Using Where Conditions and Outer Joins Together

- An Outer Join is used to select every row from the driving table, whether or not a matching row exists in the other table.
 - when there are no naturally-occurring matching rows in the other table, Oracle generates a Null-filled row to match it to.
- Using Where clauses and Outer Joins together can give unexpected results.
 - beware of the **timing** of these two operations!
 - a WHERE condition restricts the result set generated **after** the OUTER JOIN has been performed

7

Using Where Conditions and ANSI Outer Joins Together pg 426+

Show all BUS articles	<pre>SQL> SELECT title, type, ln, fn, freelancer FROM writer w RIGHT OUTER JOIN article a ON w.writerid = a.writerid WHERE type = 'BUS'</pre>	6 rows
Show all BUS articles written by a freelancer	<pre>SQL> SELECT title, type, ln, fn, freelancer FROM writer w RIGHT OUTER JOIN article a ON w.writerid = a.writerid WHERE type = 'BUS' AND freelancer = 'Y'</pre>	4 rows
Show all BUS articles written by a freelancer	<pre>SQL> SELECT title, type, ln, fn, freelancer FROM writer w RIGHT OUTER JOIN article a ON (w.writerid = a.writerid AND freelancer = 'Y') WHERE type = 'BUS'</pre>	6 rows

- When an OUTER JOIN has additional conditions that involve column(s) from the **non-driving** table, including those conditions in the ON clause causes the condition to be evaluated as the outer join is being performed, not afterward

8

Oracle Outer Join Operator (+)

- Indicates that Oracle should generate Null rows when there is no matching row in the other table
 - is used in conjunction with the **non-driving** table
- Show article titles and their descriptions, including types that don't have a matching article

```
SQL> SELECT title, descr, length
      FROM type t, article a
      WHERE a.type(+) = t.type
      ORDER BY descr;
```

- Which is the **driving** table?
- Which **rows** have NULLS?
- Which **columns** have NULLS?

9

Oracle Outer Join Operator (+) Examples

- What would each of the following produce?

```
SQL> SELECT title, ln, fn
      FROM writer w, article a
```

```
SQL> SELECT title, ln, fn
      FROM writer w, article a
      WHERE w.writerid = a.writerid
```

```
SQL> SELECT title, ln, fn
      FROM writer w, article a
      WHERE w.writerid(+) = a.writerid
```

```
SQL> SELECT title, ln, fn
      FROM writer w, article a
      WHERE w.writerid = a.writerid(+)
```

10

Using Where Conditions and Oracle Outer Join Operator(+) Together pg 424+

```
SQL> SELECT title, type, ln, fn, freelancer
      FROM writer w, article a
      WHERE w.writerid(+) = a.writerid
      AND type = 'BUS' 6 rows
```

```
SQL> SELECT title, type, ln, fn, freelancer
      FROM writer w, article a
      WHERE w.writerid(+) = a.writerid
      AND type = 'BUS'
      AND freelancer = 'Y' 4 rows
```

```
SQL> SELECT title, type, ln, fn, freelancer
      FROM writer w, article a
      WHERE w.writerid(+) = a.writerid
      AND type = 'BUS'
      AND w.freelancer(+) = 'Y' 6 rows
```

- When an OUTER JOIN has additional conditions that involve column(s) from the non-driving table, including an outer join operator on the column(s) from the **non-driving** table causes the condition to be evaluated as the outer join is being performed

11

Referential Integrity

- Requires that each foreign key in a row in the Many side table must match the primary key of a row in the One side table
 - eg: can't have an Article with an unmatched writerid
- When you define a relationship between (two) tables, referential integrity prevents unmatched rows in the Many side
 - Oracle won't permit an unmatched foreign key
 - use a Foreign Key **constraint** to enforce referential integrity (chapter 11)

```
SQL> INSERT INTO article VALUES(articlenum_seq.NEXTVAL, 'New Article', 'BUS', Null, Null, 'A000')
```

```
ERROR at line 1:
ORA-02291: integrity constraint (TTROLLEN.ARTICLE_WRITERID_FK) violated -
parent key not found
```

- Are unmatched rows allowed on the One side?
 - eg: can you have a Writer without any matching Articles?

12

Create Unmatched Rows to Demo Referential Integrity

- The SQL statements below (Chapter 10) cause **unmatched** rows in both the writer and article tables

```
SQL> ALTER TABLE article
      DISABLE CONSTRAINT article_writerid_fk;

SQL> INSERT INTO article VALUES(articlenum_seq.NEXTVAL,
'New Article', 'BUS', Null, Null, 'A000');

SQL> UPDATE ARTICLE SET writerid = 'X999'
      WHERE writerid = 'J525';

SQL> INSERT INTO writer
VALUES ('X123','New','Writer',Null,Null,Null,Null,Null);

SQL> COMMIT;
```

13

Full Outer Join

- Shows every row in each table, including:
 - rows in table A that match rows in table B
 - rows in table A that don't match a row in table B
 - rows in table B that don't match a row in table A

```
SQL> SELECT title, ln, fn
      FROM writer w FULL OUTER JOIN article a
      ON w.writerid = a.writerid
      ORDER BY title;
```

14

Full Outer Join Using Oracle Outer Join Operator (+)

- Oracle doesn't support dual (+) in a single SELECT statement
 - must use a UNION operator and two outer join operators (+)

```
SQL> SELECT title, ln, fn
      FROM writer w, article a
      WHERE w.writerid(+) = a.writerid(+)
```

```
SQL> SELECT title, ln, fn
      FROM writer w, article a
      WHERE w.writerid(+) = a.writerid
      UNION
      SELECT title, ln, fn
      FROM writer w, article a
      WHERE w.writerid = a.writerid(+)
```

15



Fix-Up Unmatched Rows

- Repair changes that violated referential integrity

```
SQL> DELETE
      FROM article
      WHERE title = 'New Article';

SQL> UPDATE article
      SET writerid = 'J525'
      WHERE writerid = 'X999';

SQL> DELETE
      FROM writer
      WHERE writerid = 'X123';

SQL> ALTER TABLE article
      ENABLE CONSTRAINT article_writerid_fk;
```

16



Self-Joins

- A join in which a column in a table references another column in the [same table](#)
- The FROM clause cites the same table twice but uses table aliases to control the context each is used
- Show each writer and their contact person/boss

```
SQL> SELECT w.ln || ', ' || w.fn as "Writer", w.freelancer,
           contact.ln || ', ' || contact.fn as Boss
      FROM writer w INNER JOIN writer contact
           ON w.contact = contact.writerid
      ORDER BY Boss, w.ln, w.fn;
```

17



Self-Joins

- Show each writer and their boss, including writers who lack a boss
 - use an OUTER JOIN to include unmatched records

```
SQL> SELECT w.ln || ', ' || w.fn as "Writer", w.freelancer,
           contact.ln || ', ' || contact.fn as Boss
      FROM writer w LEFT OUTER JOIN writer contact
           ON w.contact = contact.writerid
      ORDER BY Boss, w.ln, w.fn;
```

18
