Databases for Enterprise

Assessment Part B

S12763849

MSc Computing

January 2013

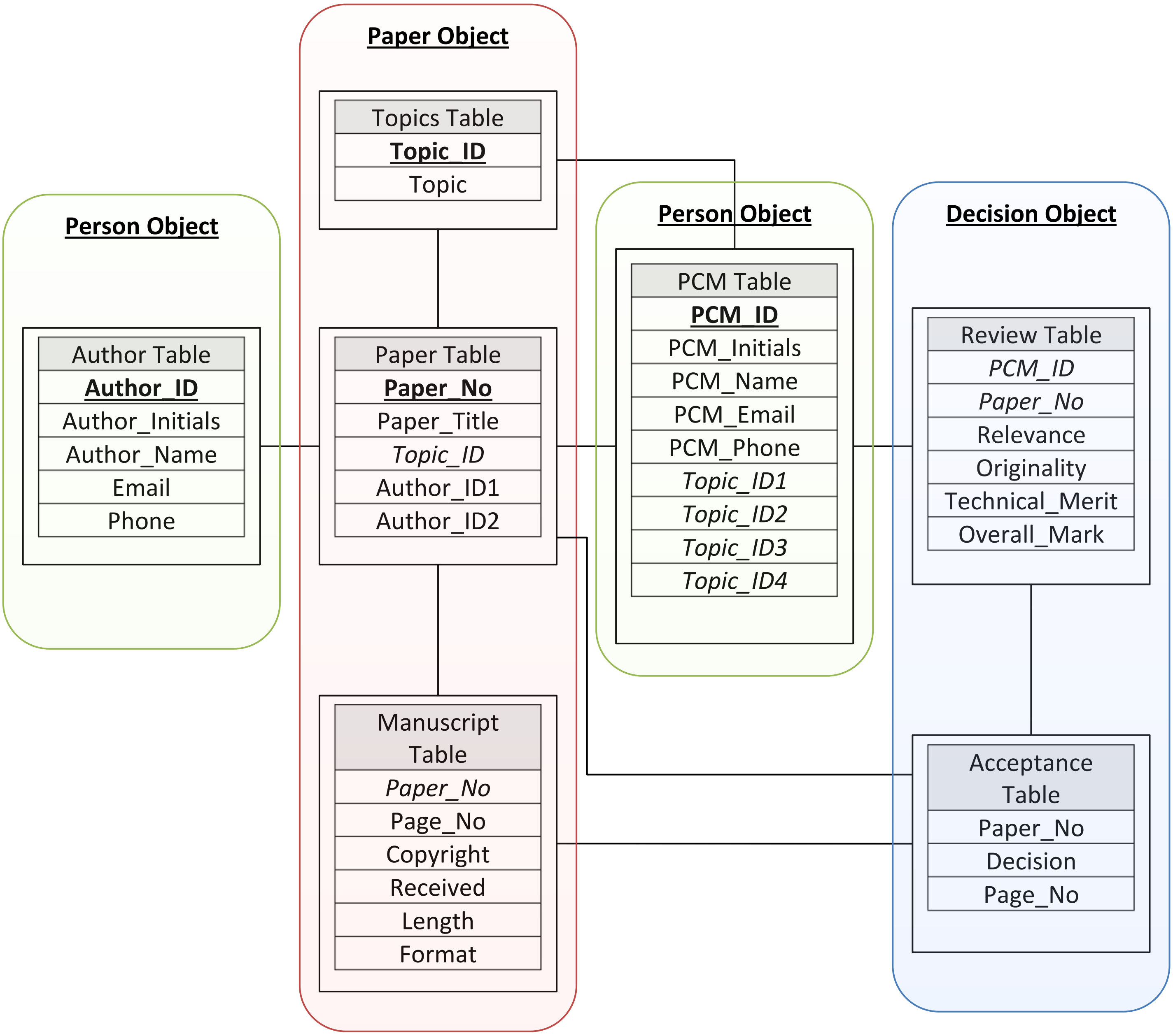
Table of Contents

Task 1

*Taking your existing tables, identify opportunities for introducing **object-oriented** data structures (including **arrays** and **nested tables**). This will mean dropping your old tables and re-building new ones with the new OO features. You will then re-populate these hybrid relational/object structures with the same or similar data from assessment A.*

Based on the original tables used in Part A of this paper, it is possible to bring in new Object Oriented features and create objects to replace the original relational database tables. As seen below new objects could be created, such as the Paper Object, the Person Object and the Decision Object.

New Objects from Previous Relational Database Model



Paper Object

Topics Table
Topic_ID
Topic

Paper Table
Paper_No
Paper_Title
Topic_ID
Author_ID1
Author_ID2

Manuscript Table
Paper_No
Page_No
Copyright
Received
Length
Format

Person Object

Author Table
Author_ID
Author_Initials
Author_Name
Email
Phone

Person Object

PCM Table
PCM_ID
PCM_Initials
PCM_Name
PCM_Email
PCM_Phone
Topic_ID1
Topic_ID2
Topic_ID3
Topic_ID4

Decision Object

Review Table
PCM_ID
Paper_No
Relevance
Originality
Technical_Merit
Overall_Mark

Acceptance Table
Paper_No
Decision
Page_No

New Objects

Person

Initials	CHAR (3),
Surname	CHAR (20),
Email	CHAR (40),
Telephone	CHAR (20));

Paper

Title	CHAR (60),
Page_No	INTEGER (3),
Copyright	CHAR (1),
Received	CHAR (1),
Length	CHAR (1),
Format	CHAR (1));

Decision

Relevance	INTEGER (2),
Originality	INETGER (2),
Technical_Merit	INTEGER (2),
Overall_Mark	INTEGER (2));

New/Edited Tables

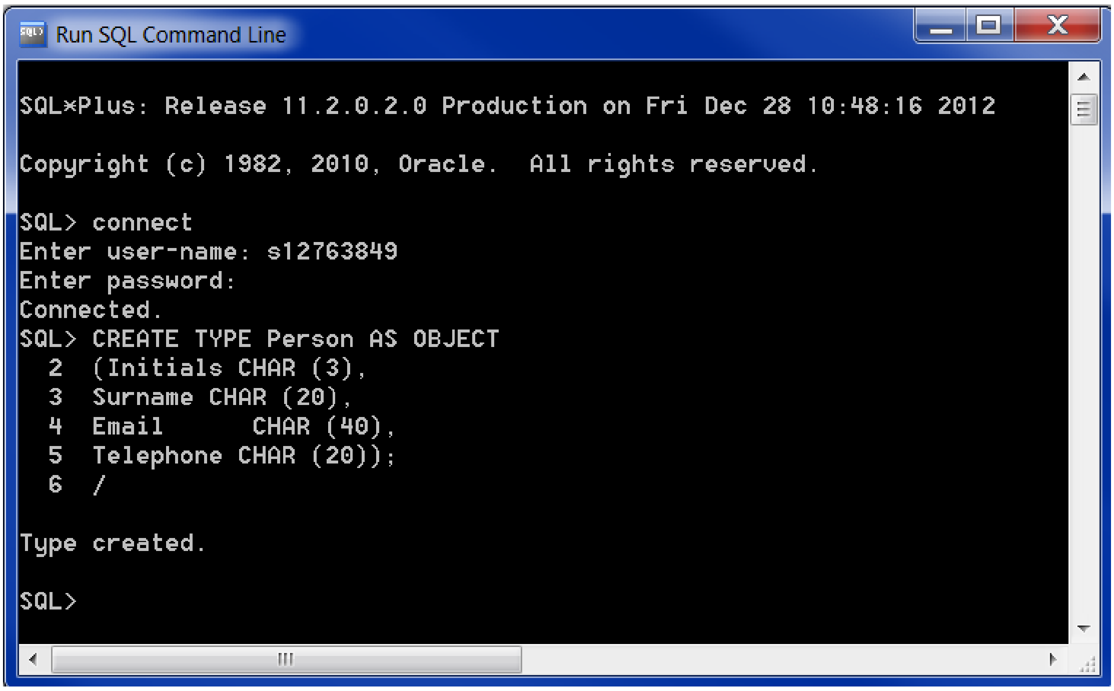
As only selected fields will be taken from the highlighted relational tables to produce the objects there will be new tables created using these objects. The new tables using Object Oriented features and functionality will therefore be;

Author	PCM	Topics	Papers	Review
Author_ID	**PCM_ID**	**Topic_ID**	**Paper_No**	*PCM_ID*
Personal Details	Personal Details	Topic	Paper Details	*Paper_No*
	Topic_ID1		Author_ID1	Decision Details
	Topic_ID2		Author_ID2	Decision (Y/N)
	Topic_ID3			
	Topic_ID4			

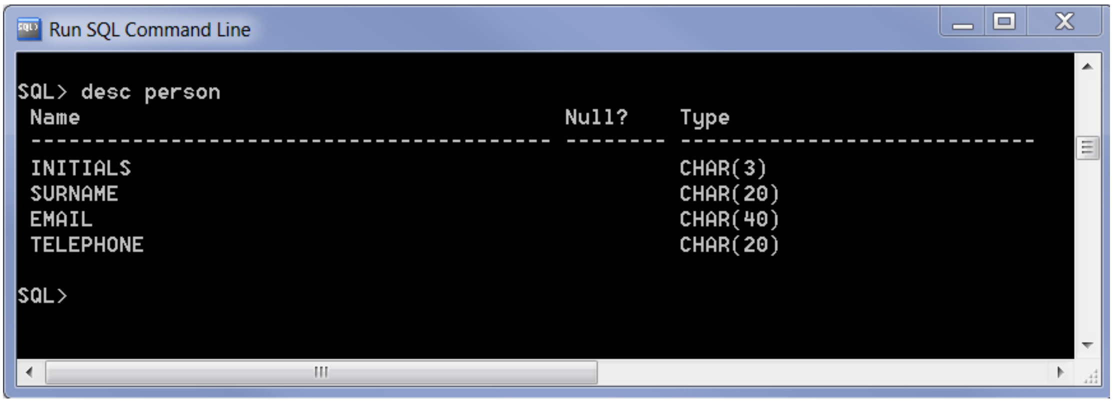
New Objects SQL and Screen Shots

Person Object

```
CREATE TYPE Person AS OBJECT
(Initials CHAR (3),
Surname CHAR (20),
Email CHAR (40),
Telephone CHAR (20));
```



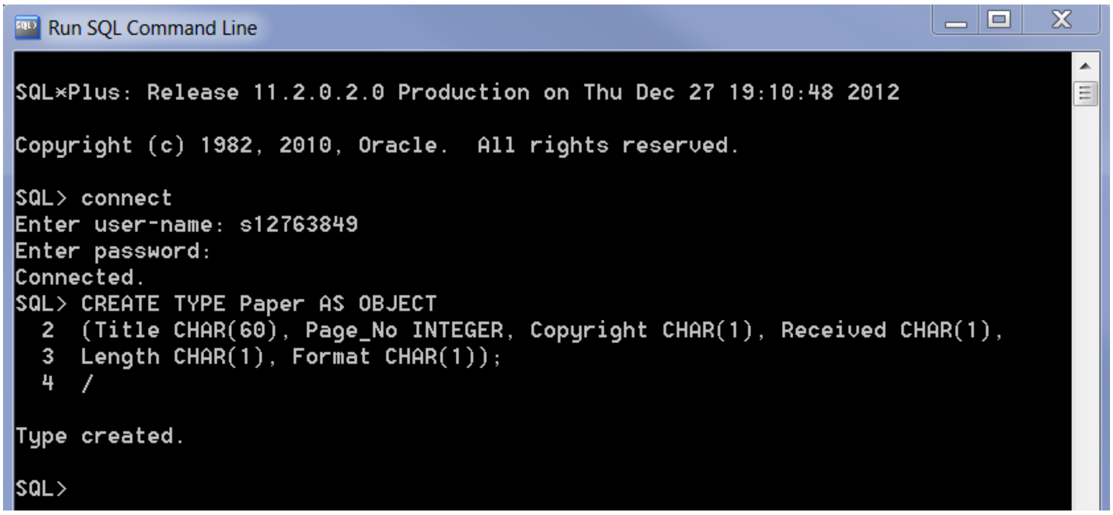
```
SQL*Plus: Release 11.2.0.2.0 Production on Fri Dec 28 10:48:16 2012

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

SQL> connect
Enter user-name: s12763849
Enter password:
Connected.
SQL> CREATE TYPE Person AS OBJECT
  2  (Initials CHAR (3),
  3  Surname CHAR (20),
  4  Email      CHAR (40),
  5  Telephone CHAR (20));
  6  /

Type created.

SQL>
```



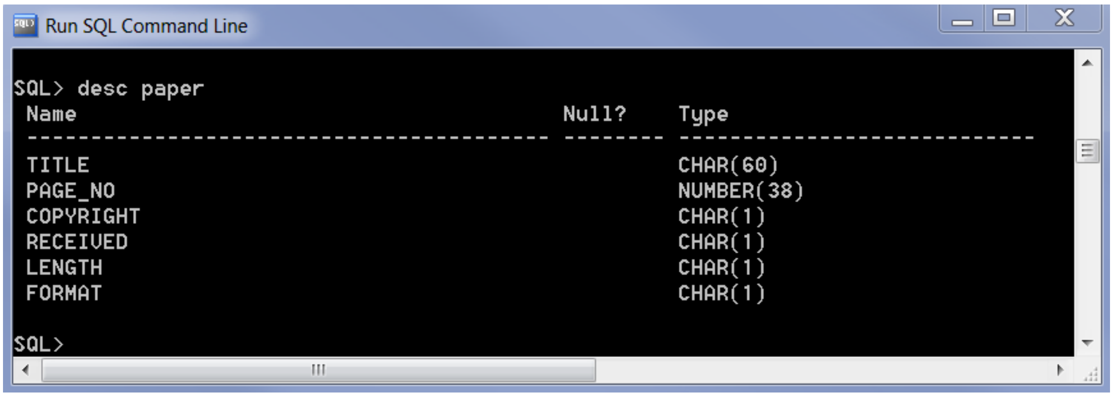
```
SQL> desc person
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 INITIALS                                           CHAR(3)
 SURNAME                                            CHAR(20)
 EMAIL                                              CHAR(40)
 TELEPHONE                                          CHAR(20)

SQL>
```

Paper Object

```
CREATE TYPE Paper AS OBJECT
(Title CHAR (60),
Page_No INTEGER,
Copyright CHAR (1),
Received CHAR (1),
Length CHAR (1),
Format CHAR (1));
```



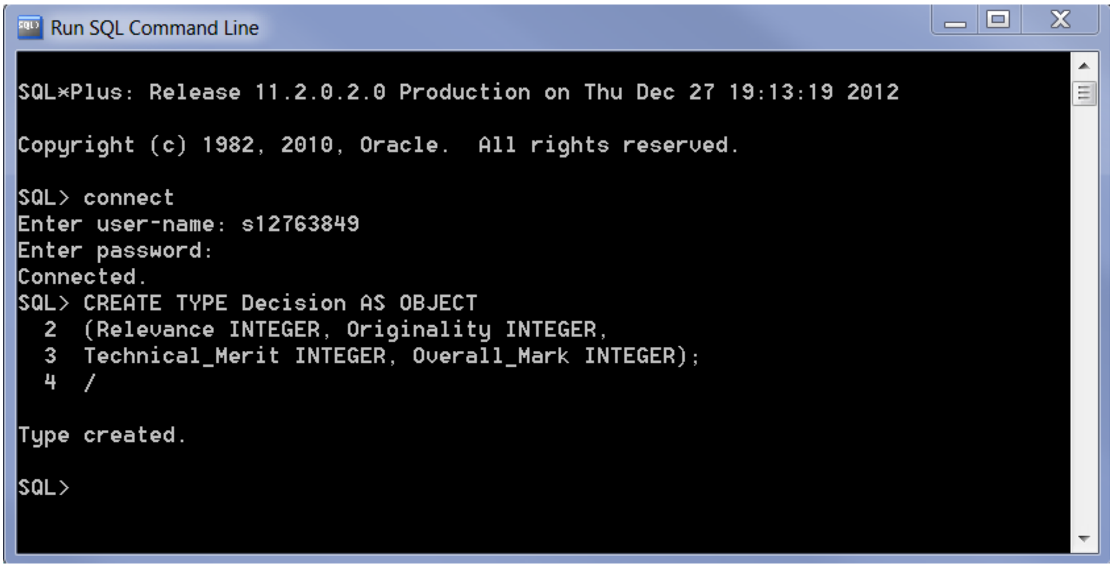
```
SQL*Plus: Release 11.2.0.2.0 Production on Thu Dec 27 19:10:48 2012

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

SQL> connect
Enter user-name: s12763849
Enter password:
Connected.
SQL> CREATE TYPE Paper AS OBJECT
  2  (Title CHAR(60), Page_No INTEGER, Copyright CHAR(1), Received CHAR(1),
  3  Length CHAR(1), Format CHAR(1));
  4  /

Type created.

SQL>
```



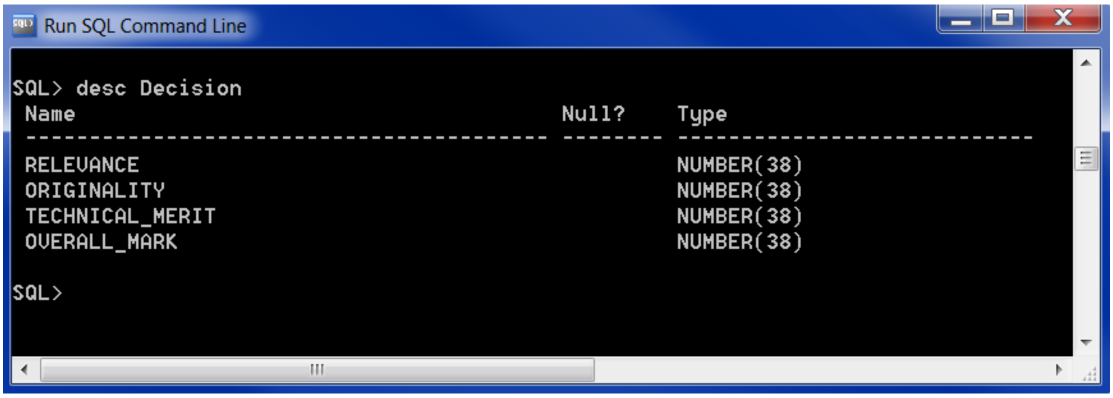
```
SQL> desc paper
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 TITLE                                              CHAR(60)
 PAGE_NO                                            NUMBER(38)
 COPYRIGHT                                          CHAR(1)
 RECEIVED                                           CHAR(1)
 LENGTH                                             CHAR(1)
 FORMAT                                             CHAR(1)

SQL>
```

Decision Object

```
CREATE TYPE Decision AS OBJECT
(Relevance          INTEGER,
Originality         INTEGER,
Technical_Merit     INTEGER,
Overall_Mark        INTEGER);
```



```
SQL*Plus: Release 11.2.0.2.0 Production on Thu Dec 27 19:13:19 2012

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

SQL> connect
Enter user-name: s12763849
Enter password:
Connected.
SQL> CREATE TYPE Decision AS OBJECT
  2  (Relevance INTEGER, Originality INTEGER,
  3  Technical_Merit INTEGER, Overall_Mark INTEGER);
  4  /

Type created.

SQL>
```

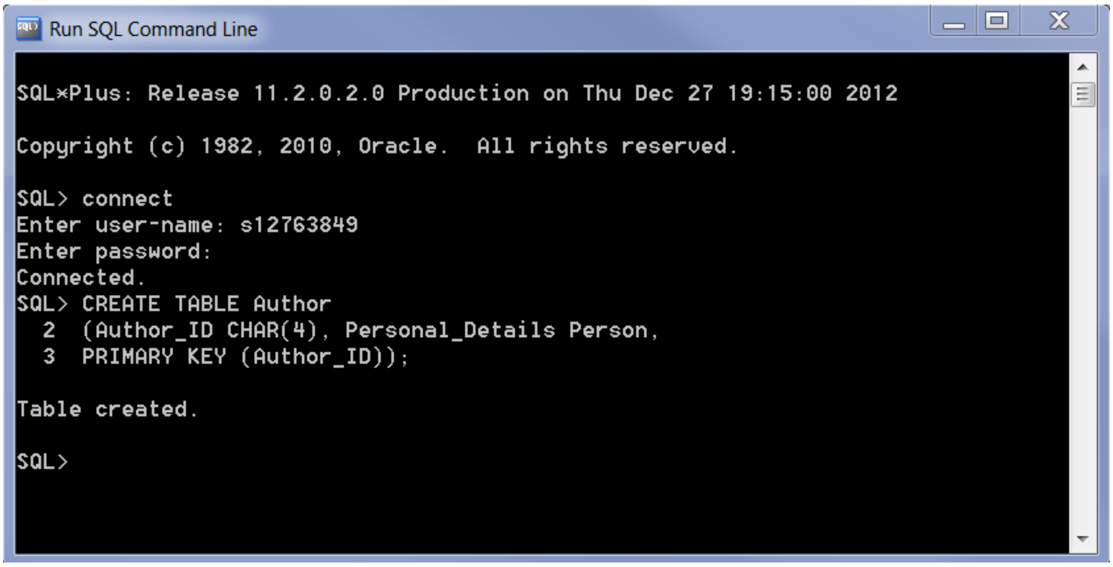


```
SQL> desc Decision
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 RELEVANCE                                          NUMBER(38)
 ORIGINALITY                                        NUMBER(38)
 TECHNICAL_MERIT                                    NUMBER(38)
 OVERALL_MARK                                       NUMBER(38)

SQL>
```

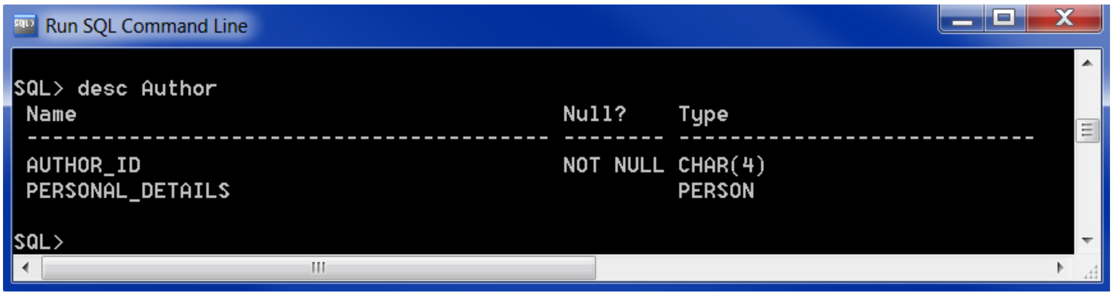
New Tables SQL and Screen Shots

Author

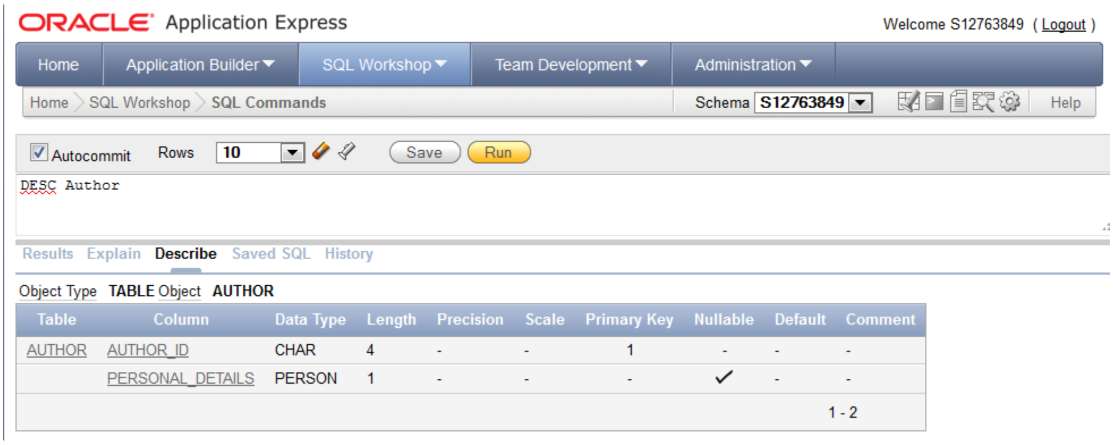
```
CREATE TABLE Author
(Author_ID CHAR (4),
Personal_Details Person,
PRIMARY KEY (Author_ID));
```



```
SQL*Plus: Release 11.2.0.2.0 Production on Thu Dec 27 19:15:00 2012

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

SQL> connect
Enter user-name: s12763849
Enter password:
Connected.
SQL> CREATE TABLE Author
  2  (Author_ID CHAR(4), Personal_Details Person,
  3  PRIMARY KEY (Author_ID));

Table created.

SQL>
```



```
SQL> desc Author
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 AUTHOR_ID                                 NOT NULL CHAR(4)
 PERSONAL_DETAILS                                   PERSON

SQL>
```



Schema S12763849

☑ Autocommit Rows 10

```
DESC Author
```

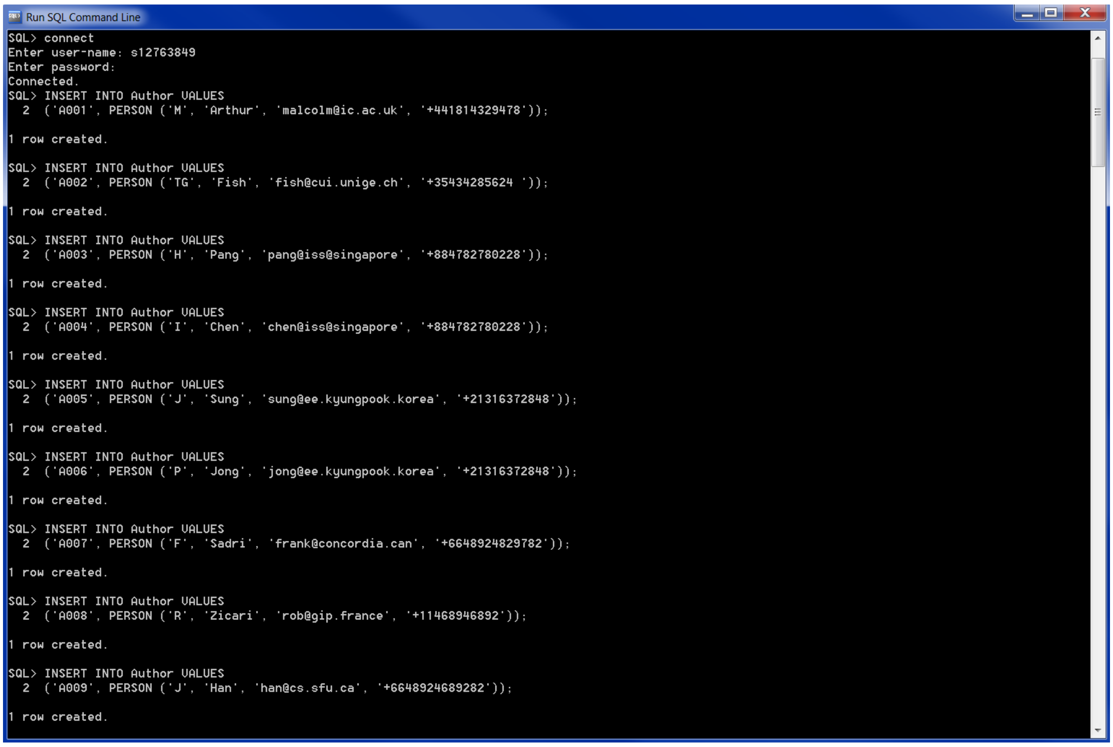
Object Type TABLE Object AUTHOR

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
AUTHOR	AUTHOR_ID	CHAR	4	-	-	1	-	-	-
	PERSONAL_DETAILS	PERSON	1	-	-	-	✓	-	-

1 - 2

Populated Table with SQL Commands

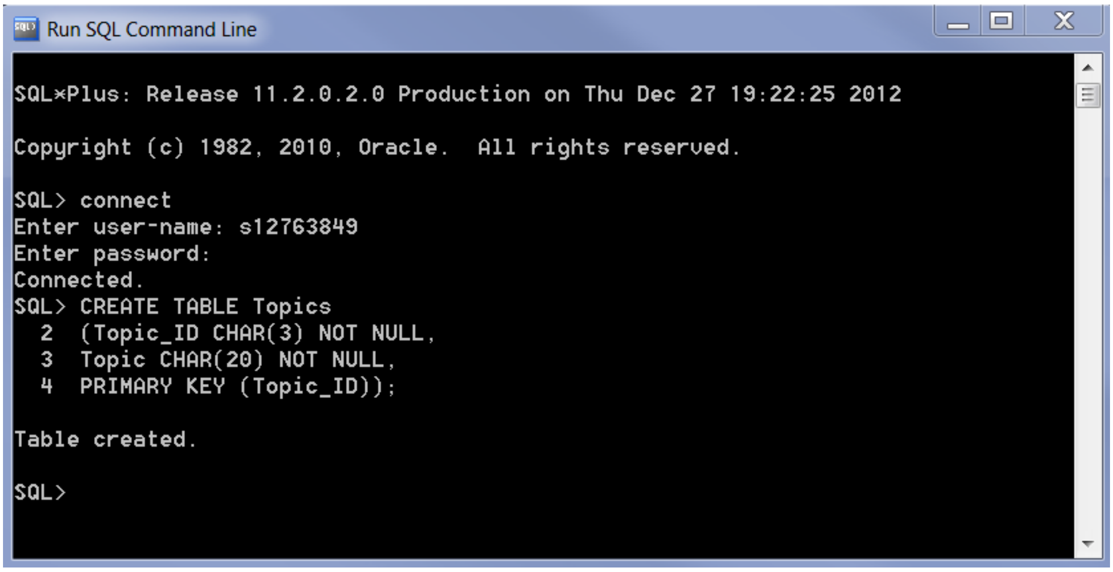
```
INSERT INTO Author VALUES
('A001', PERSON ('M', 'Arthur', 'malcolm@ic.ac.uk',
'+441814329478'));
INSERT INTO Author VALUES
('A002', PERSON ('TG', 'Fish', 'fish@cui.unige.ch', '+35434285624
'));
INSERT INTO Author VALUES
('A003', PERSON ('H', 'Pang', 'pang@iss@singapore',
'+884782780228'));
INSERT INTO Author VALUES
('A004', PERSON ('I', 'Chen', 'chen@iss@singapore',
'+884782780228'));
INSERT INTO Author VALUES
('A005', PERSON ('J', 'Sung', 'sung@ee.kyungpook.korea',
'+21316372848'));
INSERT INTO Author VALUES
('A006', PERSON ('P', 'Jong', 'jong@ee.kyungpook.korea',
'+21316372848'));
INSERT INTO Author VALUES
('A007', PERSON ('F', 'Sadri', 'frank@concordia.can',
'+6648924829782'));
INSERT INTO Author VALUES
('A008', PERSON ('R', 'Zicari', 'rob@gip.france', '+11468946892'));
INSERT INTO Author VALUES
('A009', PERSON ('J', 'Han', 'han@cs.sfu.ca', '+6648924689282'));
```



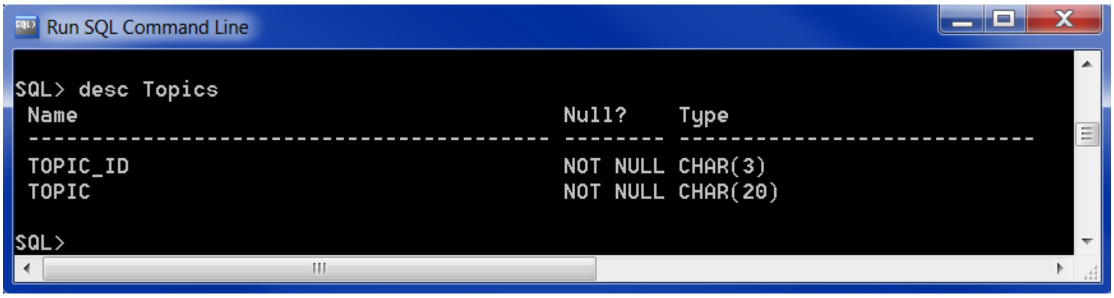
```
SQL> connect
Enter user-name: s12763849
Enter password:
Connected.
SQL> INSERT INTO Author VALUES
  2  ('A001', PERSON ('M', 'Arthur', 'malcolm@ic.ac.uk', '+441814329478'));

1 row created.

SQL> INSERT INTO Author VALUES
  2  ('A002', PERSON ('TG', 'Fish', 'fish@cui.unige.ch', '+35434285624 '));

1 row created.

SQL> INSERT INTO Author VALUES
  2  ('A003', PERSON ('H', 'Pang', 'pang@iss@singapore', '+884782780228'));

1 row created.

SQL> INSERT INTO Author VALUES
  2  ('A004', PERSON ('I', 'Chen', 'chen@iss@singapore', '+884782780228'));

1 row created.

SQL> INSERT INTO Author VALUES
  2  ('A005', PERSON ('J', 'Sung', 'sung@ee.kyungpook.korea', '+21316372848'));

1 row created.

SQL> INSERT INTO Author VALUES
  2  ('A006', PERSON ('P', 'Jong', 'jong@ee.kyungpook.korea', '+21316372848'));

1 row created.

SQL> INSERT INTO Author VALUES
  2  ('A007', PERSON ('F', 'Sadri', 'frank@concordia.can', '+6648924829782'));

1 row created.

SQL> INSERT INTO Author VALUES
  2  ('A008', PERSON ('R', 'Zicari', 'rob@gip.france', '+11468946892'));

1 row created.

SQL> INSERT INTO Author VALUES
  2  ('A009', PERSON ('J', 'Han', 'han@cs.sfu.ca', '+6648924689282'));

1 row created.
```

Topics

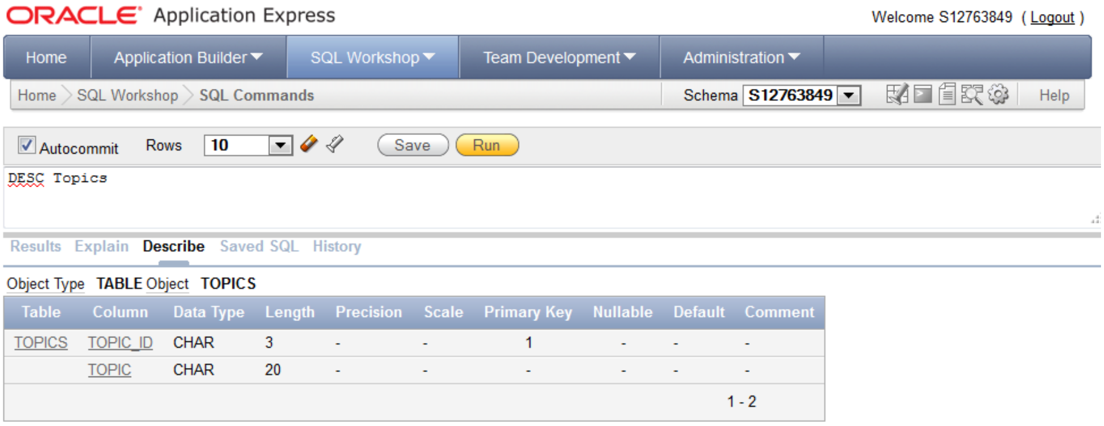
```
CREATE TABLE Topics
(Topic_ID CHAR (3) NOT NULL,
Topic CHAR (20) NOT NULL,
PRIMARY KEY (Topic_ID));
```



```
SQL*Plus: Release 11.2.0.2.0 Production on Thu Dec 27 19:22:25 2012

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

SQL> connect
Enter user-name: s12763849
Enter password:
Connected.
SQL> CREATE TABLE Topics
  2  (Topic_ID CHAR(3) NOT NULL,
  3  Topic CHAR(20) NOT NULL,
  4  PRIMARY KEY (Topic_ID));

Table created.

SQL>
```



```
SQL> desc Topics
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 TOPIC_ID                                  NOT NULL CHAR(3)
 TOPIC                                     NOT NULL CHAR(20)

SQL>
```



Schema S12763849

☑ Autocommit Rows 10

```
DESC Topics
```

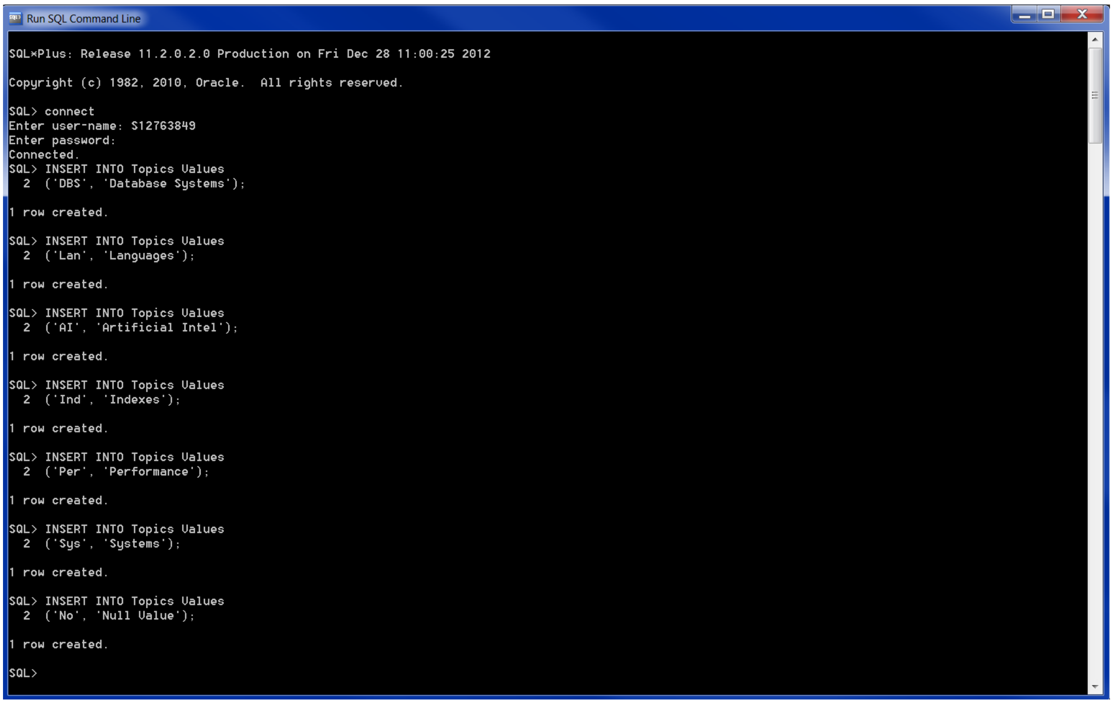
Object Type TABLE Object TOPICS

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
TOPICS	TOPIC_ID	CHAR	3	-	-	1	-	-	-
	TOPIC	CHAR	20	-	-	-	-	-	-

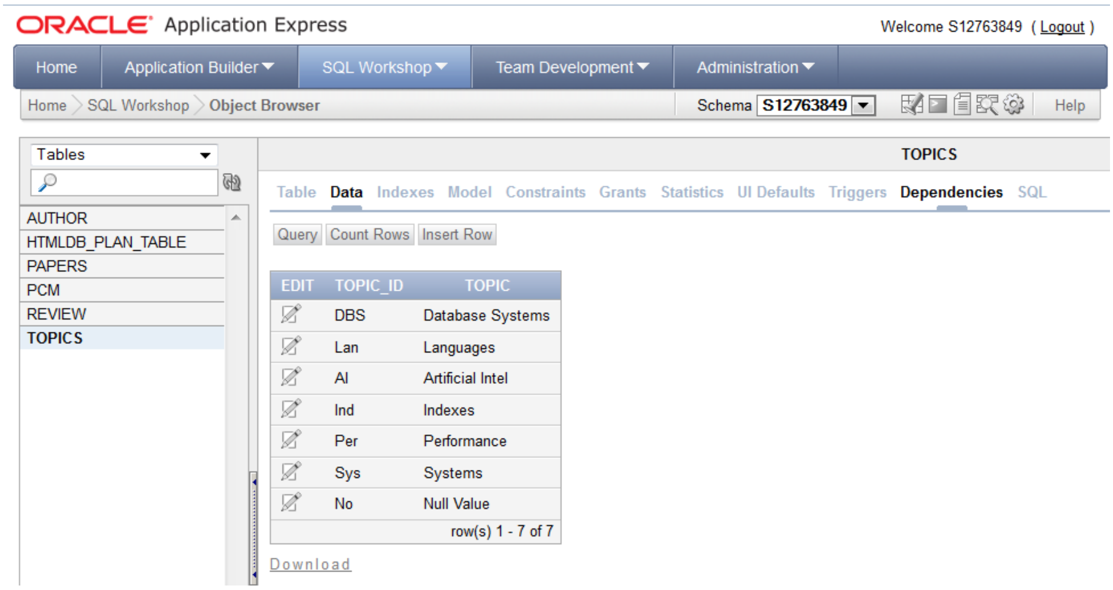
1 - 2

Populated Table with SQL Commands

```
INSERT INTO Topics Values
('DBS', 'Database Systems');
INSERT INTO Topics Values
('Lan', 'Languages');
INSERT INTO Topics Values
('AI', 'Artificial Intel');
INSERT INTO Topics Values
('Ind', 'Indexes');
INSERT INTO Topics Values
('Per', 'Performance');
INSERT INTO Topics Values
('Sys', 'Systems');
INSERT INTO Topics Values
('No', 'Null Value');
```



```
SQL*Plus: Release 11.2.0.2.0 Production on Fri Dec 28 11:00:25 2012

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

SQL> connect
Enter user-name: S12763849
Enter password:
Connected.
SQL> INSERT INTO Topics Values
  2  ('DBS', 'Database Systems');

1 row created.

SQL> INSERT INTO Topics Values
  2  ('Lan', 'Languages');

1 row created.

SQL> INSERT INTO Topics Values
  2  ('AI', 'Artificial Intel');

1 row created.

SQL> INSERT INTO Topics Values
  2  ('Ind', 'Indexes');

1 row created.

SQL> INSERT INTO Topics Values
  2  ('Per', 'Performance');

1 row created.

SQL> INSERT INTO Topics Values
  2  ('Sys', 'Systems');

1 row created.

SQL> INSERT INTO Topics Values
  2  ('No', 'Null Value');

1 row created.

SQL>
```

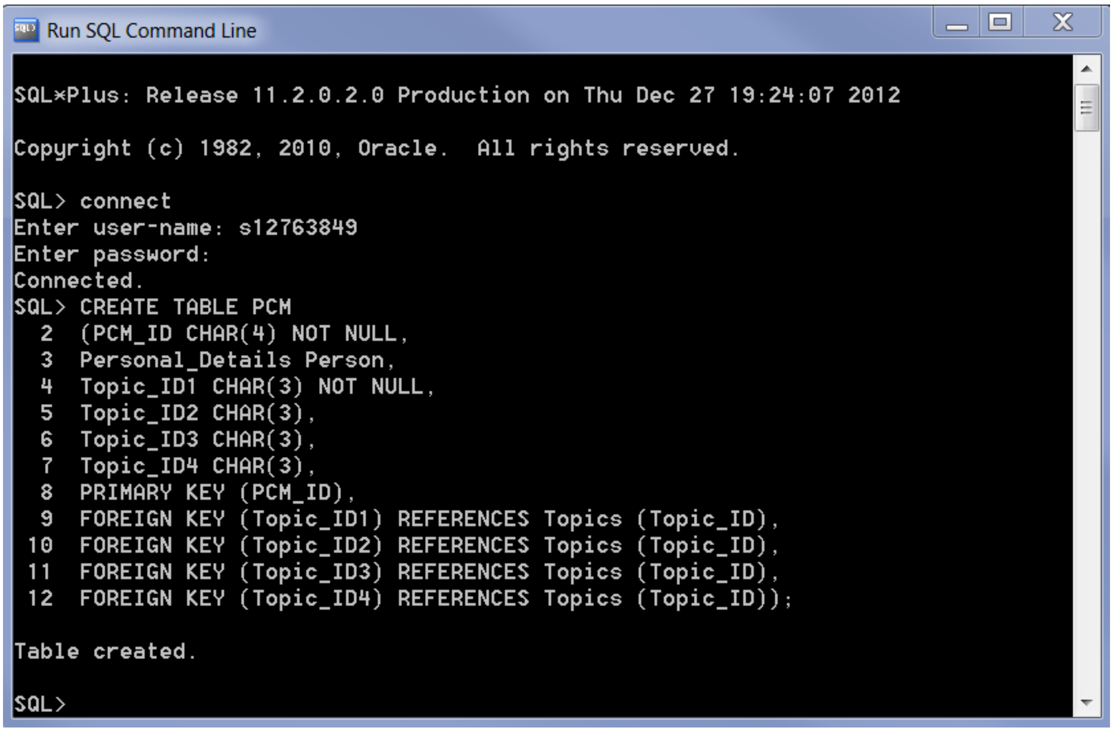


EDIT	TOPIC_ID	TOPIC
✎	DBS	Database Systems
✎	Lan	Languages
✎	AI	Artificial Intel
✎	Ind	Indexes
✎	Per	Performance
✎	Sys	Systems
✎	No	Null Value

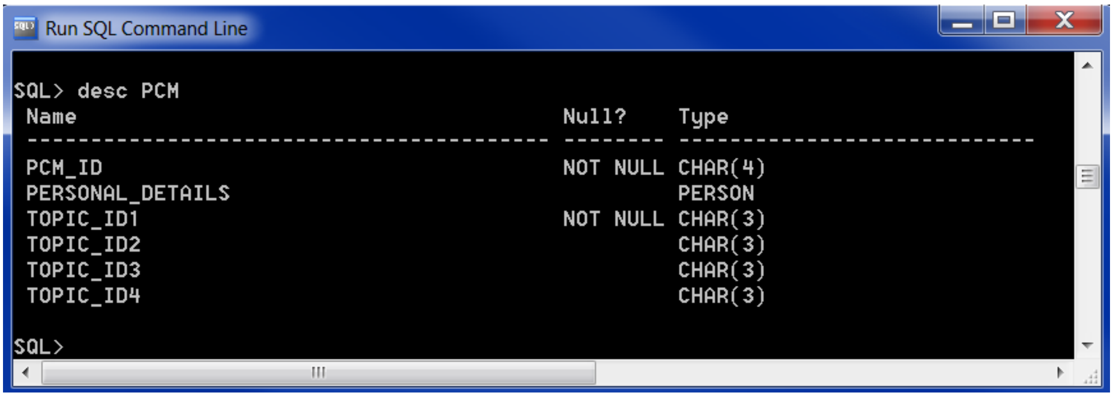
row(s) 1 - 7 of 7

PCM

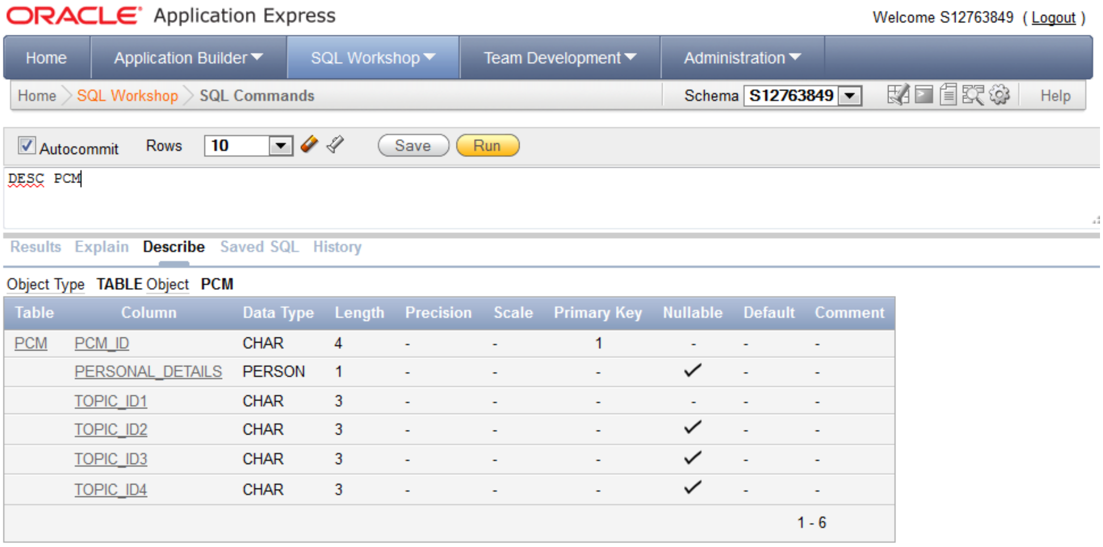
```
CREATE TABLE PCM
(PCM_ID CHAR (4) NOT NULL,
Personal_Details Person,
Topic_ID1 CHAR (3) NOT NULL,
Topic_ID2 CHAR (3),
Topic_ID3 CHAR (3),
Topic_ID4 CHAR (3),
PRIMARY KEY (PCM_ID),
FOREIGN KEY (Topic_ID1) REFERENCES Topics (Topic_ID),
FOREIGN KEY (Topic_ID2) REFERENCES Topics (Topic_ID),
FOREIGN KEY (Topic_ID3) REFERENCES Topics (Topic_ID),
FOREIGN KEY (Topic_ID4) REFERENCES Topics (Topic_ID));
```



```
SQL*Plus: Release 11.2.0.2.0 Production on Thu Dec 27 19:24:07 2012

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

SQL> connect
Enter user-name: s12763849
Enter password:
Connected.
SQL> CREATE TABLE PCM
  2  (PCM_ID CHAR(4) NOT NULL,
  3  Personal_Details Person,
  4  Topic_ID1 CHAR(3) NOT NULL,
  5  Topic_ID2 CHAR(3),
  6  Topic_ID3 CHAR(3),
  7  Topic_ID4 CHAR(3),
  8  PRIMARY KEY (PCM_ID),
  9  FOREIGN KEY (Topic_ID1) REFERENCES Topics (Topic_ID),
 10  FOREIGN KEY (Topic_ID2) REFERENCES Topics (Topic_ID),
 11  FOREIGN KEY (Topic_ID3) REFERENCES Topics (Topic_ID),
 12  FOREIGN KEY (Topic_ID4) REFERENCES Topics (Topic_ID));

Table created.

SQL>
```



```
SQL> desc PCM
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 PCM_ID                                    NOT NULL CHAR(4)
 PERSONAL_DETAILS                                   PERSON
 TOPIC_ID1                                 NOT NULL CHAR(3)
 TOPIC_ID2                                          CHAR(3)
 TOPIC_ID3                                          CHAR(3)
 TOPIC_ID4                                          CHAR(3)

SQL>
```



☑ Autocommit Rows 10

```
DESC PCM
```

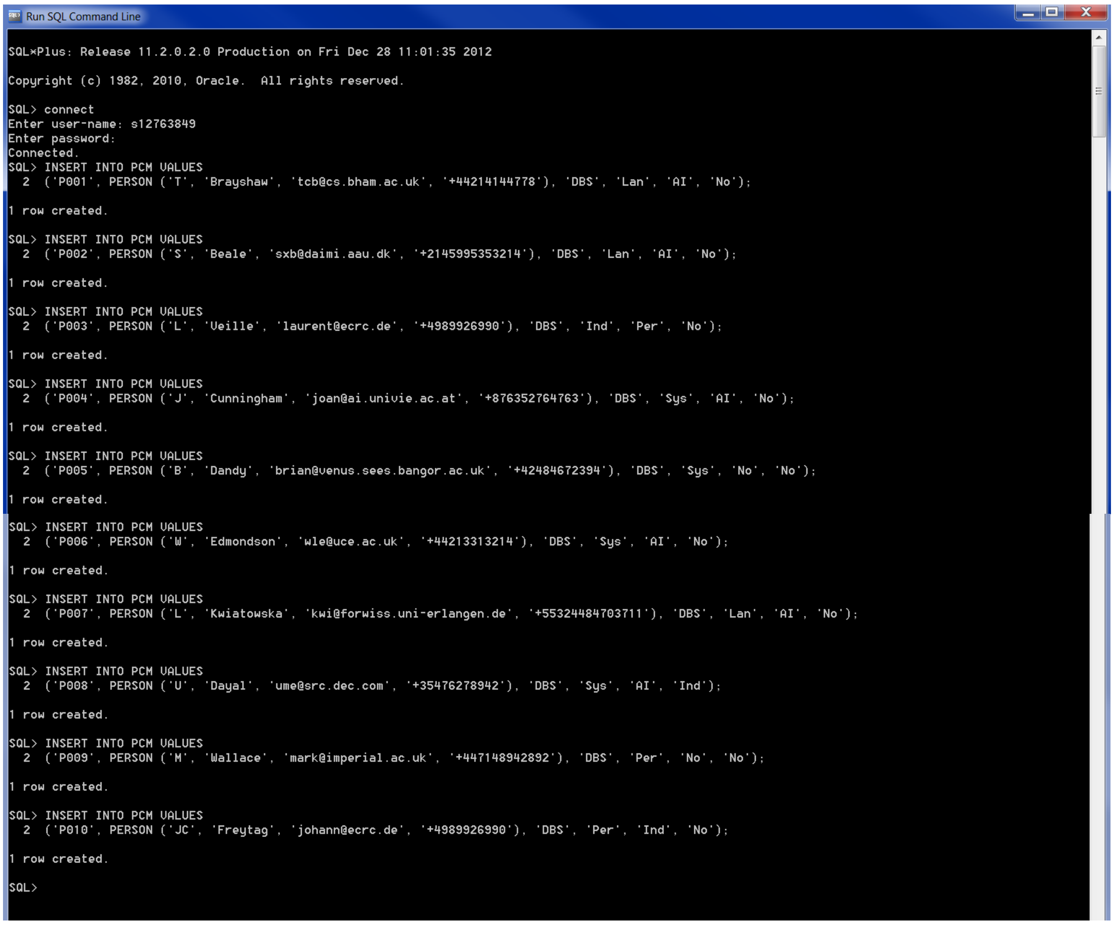
Object Type TABLE Object PCM

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PCM	PCM_ID	CHAR	4	-	-	1	-	-	-
	PERSONAL_DETAILS	PERSON	1	-	-	-	✓	-	-
	TOPIC_ID1	CHAR	3	-	-	-	-	-	-
	TOPIC_ID2	CHAR	3	-	-	-	✓	-	-
	TOPIC_ID3	CHAR	3	-	-	-	✓	-	-
	TOPIC_ID4	CHAR	3	-	-	-	✓	-	-

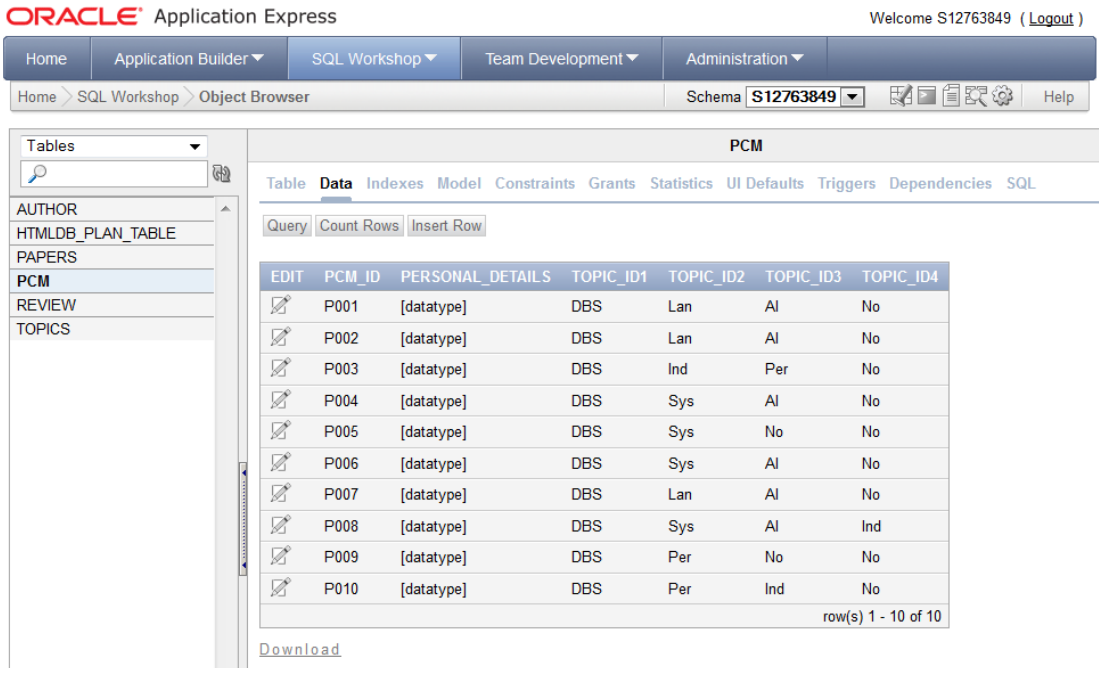
1 - 6

Populated Table with SQL Commands

```
INSERT INTO PCM VALUES
('P001', PERSON ('T', 'Brayshaw', 'tcb@cs.bham.ac.uk',
'+44214144778'), 'DBS', 'Lan', 'AI', 'No');
INSERT INTO PCM VALUES
('P002', PERSON ('S', 'Beale', 'sxb@daimi.aau.dk', '+2145995353214'),
'DBS', 'Lan', 'AI', 'No');
INSERT INTO PCM VALUES
('P003', PERSON ('L', 'Veille', 'laurent@ecrc.de', '+4989926990'),
'DBS', 'Ind', 'Per', 'No');
INSERT INTO PCM VALUES
('P004', PERSON ('J', 'Cunningham', 'joan@ai.univie.ac.at',
'+876352764763'), 'DBS', 'Sys', 'AI', 'No');
INSERT INTO PCM VALUES
('P005', PERSON ('B', 'Dandy', 'brian@venus.sees.bangor.ac.uk',
'+42484672394'), 'DBS', 'Sys', 'No', 'No');
INSERT INTO PCM VALUES
('P006', PERSON ('W', 'Edmondson', 'wle@uce.ac.uk', '+44213313214'),
'DBS', 'Sys', 'AI', 'No');
INSERT INTO PCM VALUES
('P007', PERSON ('L', 'Kwiatowska', 'kwi@forwiss.uni-erlangen.de',
'+55324484703711'), 'DBS', 'Lan', 'AI', 'No');
INSERT INTO PCM VALUES
('P008', PERSON ('U', 'Dayal', 'ume@src.dec.com', '+35476278942'),
'DBS', 'Sys', 'AI', 'Ind');
INSERT INTO PCM VALUES
('P009', PERSON ('M', 'Wallace', 'mark@imperial.ac.uk',
'+447148942892'), 'DBS', 'Per', 'No', 'No');
INSERT INTO PCM VALUES
('P010', PERSON ('JC', 'Freytag', 'johann@ecrc.de', '+4989926990'),
'DBS', 'Per', 'Ind', 'No');
```



```
SQL*Plus: Release 11.2.0.2.0 Production on Fri Dec 28 11:01:35 2012

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

SQL> connect
Enter user-name: s12763849
Enter password:
Connected.
SQL> INSERT INTO PCM VALUES
  2  ('P001', PERSON ('T', 'Brayshaw', 'tcb@cs.bham.ac.uk', '+44214144778'), 'DBS', 'Lan', 'AI', 'No');

1 row created.

SQL> INSERT INTO PCM VALUES
  2  ('P002', PERSON ('S', 'Beale', 'sxb@daimi.aau.dk', '+2145995353214'), 'DBS', 'Lan', 'AI', 'No');

1 row created.

SQL> INSERT INTO PCM VALUES
  2  ('P003', PERSON ('L', 'Veille', 'laurent@ecrc.de', '+4989926990'), 'DBS', 'Ind', 'Per', 'No');

1 row created.

SQL> INSERT INTO PCM VALUES
  2  ('P004', PERSON ('J', 'Cunningham', 'joan@ai.univie.ac.at', '+876352764763'), 'DBS', 'Sys', 'AI', 'No');

1 row created.

SQL> INSERT INTO PCM VALUES
  2  ('P005', PERSON ('B', 'Dandy', 'brian@venus.sees.bangor.ac.uk', '+42484672394'), 'DBS', 'Sys', 'No', 'No');

1 row created.

SQL> INSERT INTO PCM VALUES
  2  ('P006', PERSON ('W', 'Edmondson', 'wle@uce.ac.uk', '+44213313214'), 'DBS', 'Sys', 'AI', 'No');

1 row created.

SQL> INSERT INTO PCM VALUES
  2  ('P007', PERSON ('L', 'Kwiatowska', 'kwi@forwiss.uni-erlangen.de', '+55324484703711'), 'DBS', 'Lan', 'AI', 'No');

1 row created.

SQL> INSERT INTO PCM VALUES
  2  ('P008', PERSON ('U', 'Dayal', 'ume@src.dec.com', '+35476278942'), 'DBS', 'Sys', 'AI', 'Ind');

1 row created.

SQL> INSERT INTO PCM VALUES
  2  ('P009', PERSON ('M', 'Wallace', 'mark@imperial.ac.uk', '+447148942892'), 'DBS', 'Per', 'No', 'No');

1 row created.

SQL> INSERT INTO PCM VALUES
  2  ('P010', PERSON ('JC', 'Freytag', 'johann@ecrc.de', '+4989926990'), 'DBS', 'Per', 'Ind', 'No');

1 row created.

SQL>
```

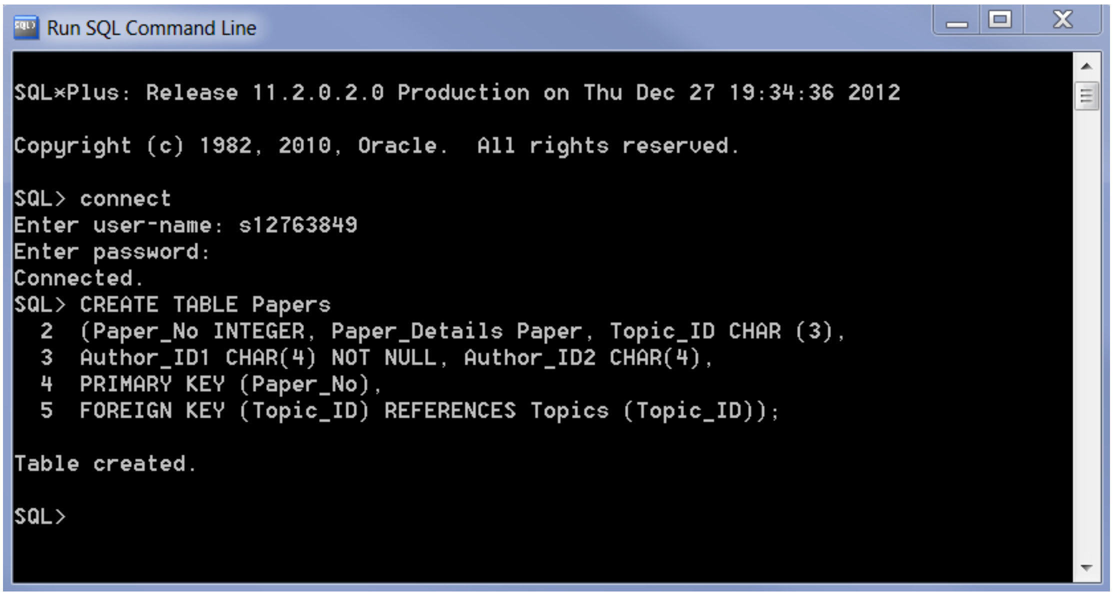


EDIT	PCM_ID	PERSONAL_DETAILS	TOPIC_ID1	TOPIC_ID2	TOPIC_ID3	TOPIC_ID4
	P001	[datatype]	DBS	Lan	AI	No
	P002	[datatype]	DBS	Lan	AI	No
	P003	[datatype]	DBS	Ind	Per	No
	P004	[datatype]	DBS	Sys	AI	No
	P005	[datatype]	DBS	Sys	No	No
	P006	[datatype]	DBS	Sys	AI	No
	P007	[datatype]	DBS	Lan	AI	No
	P008	[datatype]	DBS	Sys	AI	Ind
	P009	[datatype]	DBS	Per	No	No
	P010	[datatype]	DBS	Per	Ind	No

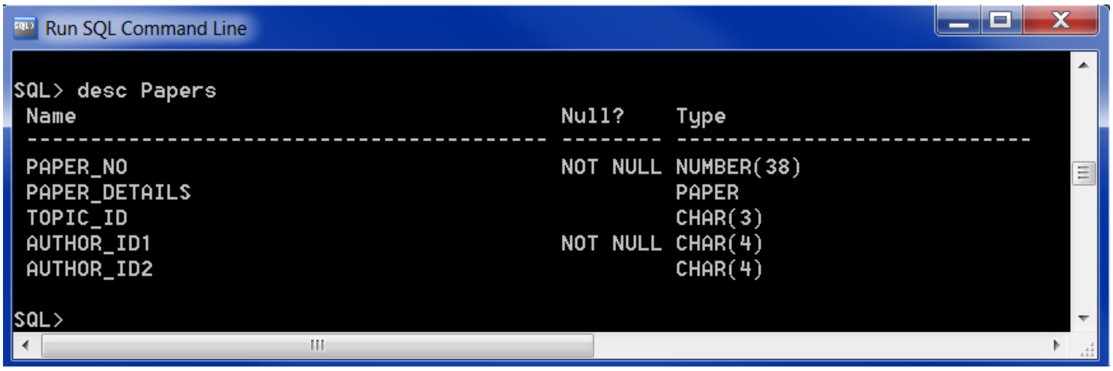
row(s) 1 - 10 of 10

Papers

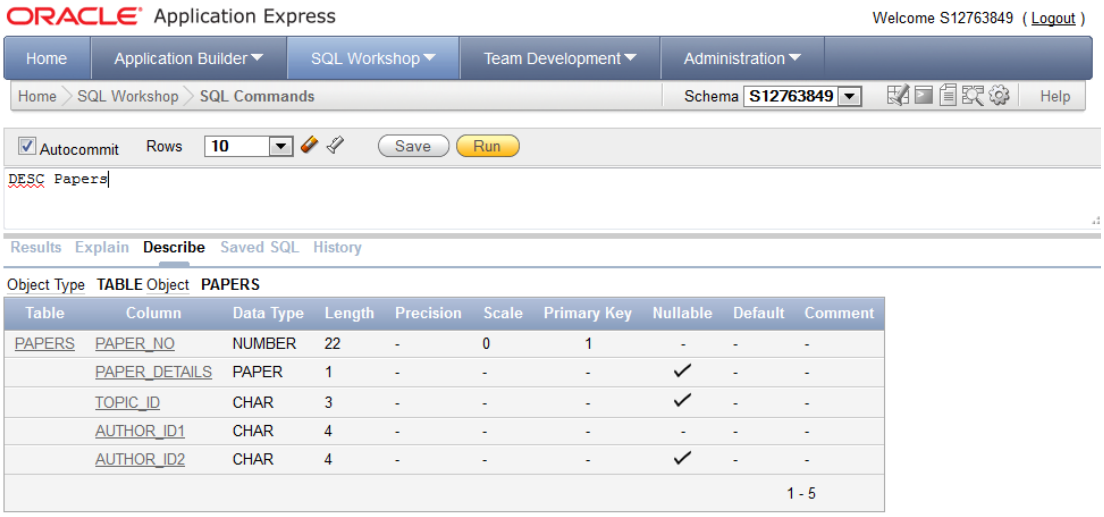
```
CREATE TABLE Papers
(Paper_No INTEGER,
Paper_Details Paper,
Topic_ID CHAR (3),
Author_ID1 CHAR (4) NOT NULL,
Author_ID2 CHAR (4),
PRIMARY KEY (Paper_No),
FOREIGN KEY (Topic_ID) REFERENCES Topics (Topic_ID));
```



```
SQL*Plus: Release 11.2.0.2.0 Production on Thu Dec 27 19:34:36 2012

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

SQL> connect
Enter user-name: s12763849
Enter password:
Connected.
SQL> CREATE TABLE Papers
     (Paper_No INTEGER, Paper_Details Paper, Topic_ID CHAR (3),
     Author_ID1 CHAR(4) NOT NULL, Author_ID2 CHAR(4),
     PRIMARY KEY (Paper_No),
     FOREIGN KEY (Topic_ID) REFERENCES Topics (Topic_ID));

Table created.

SQL>
```



```
SQL> desc Papers
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 PAPER_NO                                  NOT NULL NUMBER(38)
 PAPER_DETAILS                                      PAPER
 TOPIC_ID                                           CHAR(3)
 AUTHOR_ID1                                NOT NULL CHAR(4)
 AUTHOR_ID2                                         CHAR(4)

SQL>
```



☑ Autocommit Rows 10

```
DESC Papers
```

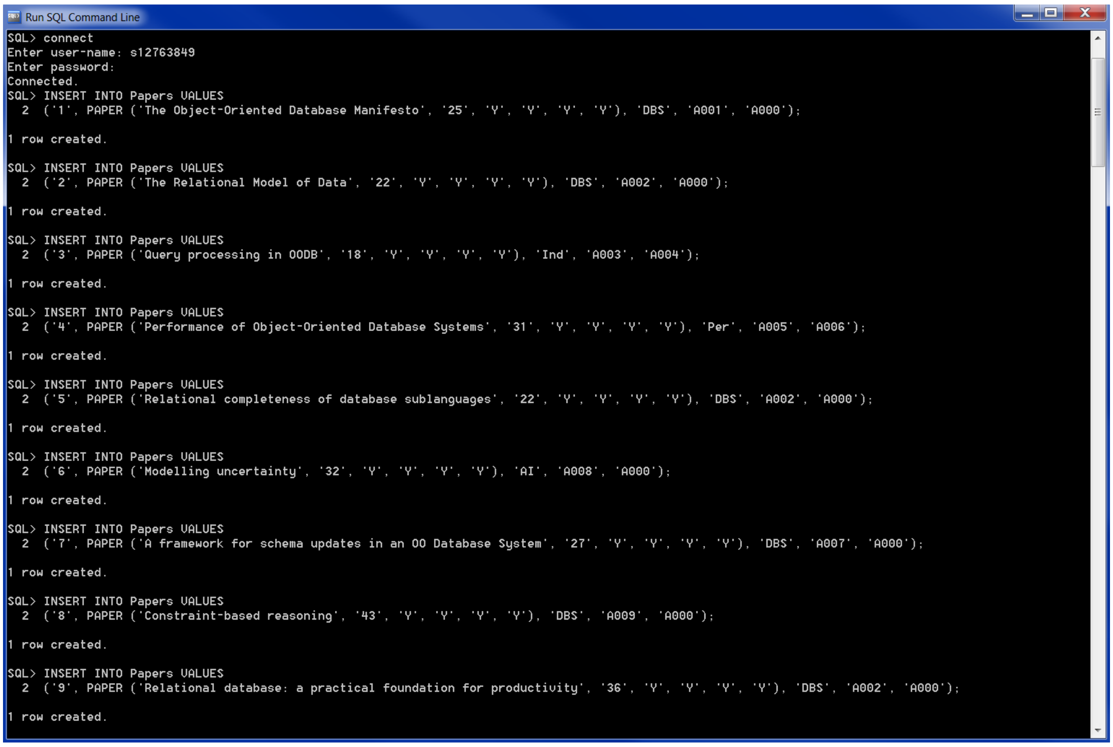
Object Type TABLE Object PAPERS

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PAPERS	PAPER_NO	NUMBER	22	-	0	1	-	-	-
	PAPER_DETAILS	PAPER	1	-	-	-	✓	-	-
	TOPIC_ID	CHAR	3	-	-	-	✓	-	-
	AUTHOR_ID1	CHAR	4	-	-	-	-	-	-
	AUTHOR_ID2	CHAR	4	-	-	-	✓	-	-

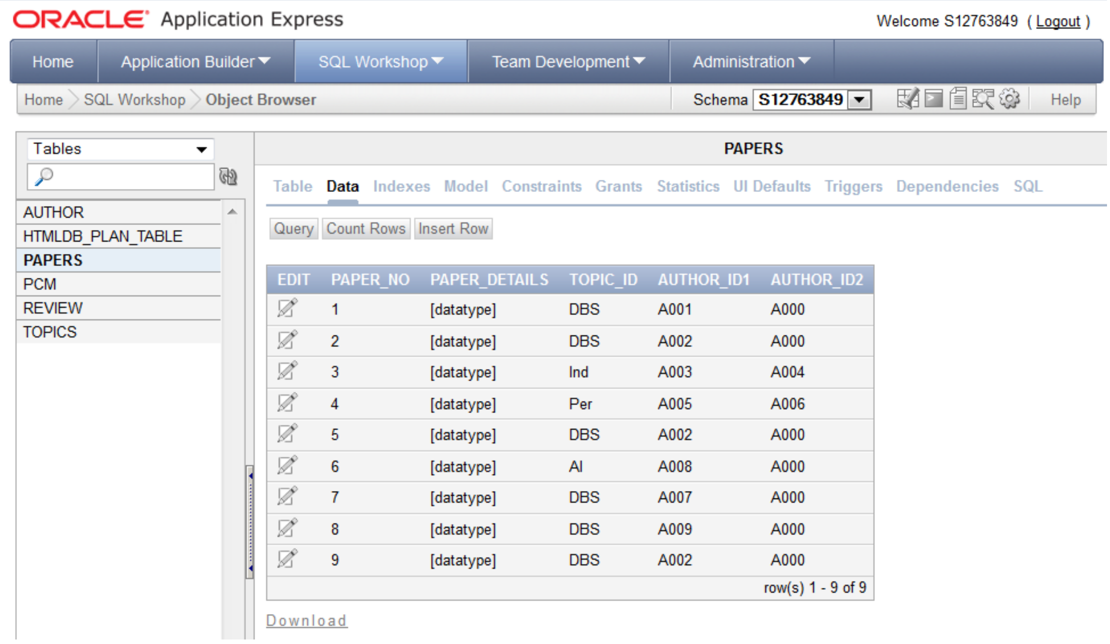
1 - 5

Populated Table with SQL Commands

```
INSERT INTO Papers VALUES
('1', PAPER ('The Object-Oriented Database Manifesto', '25', 'Y',
'Y', 'Y', 'Y'), 'DBS', 'A001', 'A000');
INSERT INTO Papers VALUES
('2', PAPER ('The Relational Model of Data', '22', 'Y', 'Y', 'Y',
'Y'), 'DBS', 'A002', 'A000');
INSERT INTO Papers VALUES
('3', PAPER ('Query processing in OODB', '18', 'Y', 'Y', 'Y', 'Y'),
'Ind', 'A003', 'A004');
INSERT INTO Papers VALUES
('4', PAPER ('Performance of Object-Oriented Database Systems', '31',
'Y', 'Y', 'Y', 'Y'), 'Per', 'A005', 'A006');
INSERT INTO Papers VALUES
('5', PAPER ('Relational completeness of database sublanguages',
'22', 'Y', 'Y', 'Y', 'Y'), 'DBS', 'A002', 'A000');
INSERT INTO Papers VALUES
('6', PAPER ('Modelling uncertainty', '32', 'Y', 'Y', 'Y', 'Y'),
'AI', 'A008', 'A000');
INSERT INTO Papers VALUES
('7', PAPER ('A framework for schema updates in an OO Database
System', '27', 'Y', 'Y', 'Y', 'Y'), 'DBS', 'A007', 'A000');
INSERT INTO Papers VALUES
('8', PAPER ('Constraint-based reasoning', '43', 'Y', 'Y', 'Y', 'Y'),
'DBS', 'A009', 'A000');
INSERT INTO Papers VALUES
('9', PAPER ('Relational database: a practical foundation for
productivity', '36', 'Y', 'Y', 'Y', 'Y'), 'DBS', 'A002', 'A000');
```



```
SQL> connect
Enter user-name: s12763849
Enter password:
Connected.
SQL> INSERT INTO Papers VALUES
  2  ('1', PAPER ('The Object-Oriented Database Manifesto', '25', 'Y', 'Y', 'Y', 'Y'), 'DBS', 'A001', 'A000');

1 row created.

SQL> INSERT INTO Papers VALUES
  2  ('2', PAPER ('The Relational Model of Data', '22', 'Y', 'Y', 'Y', 'Y'), 'DBS', 'A002', 'A000');

1 row created.

SQL> INSERT INTO Papers VALUES
  2  ('3', PAPER ('Query processing in OODB', '18', 'Y', 'Y', 'Y', 'Y'), 'Ind', 'A003', 'A004');

1 row created.

SQL> INSERT INTO Papers VALUES
  2  ('4', PAPER ('Performance of Object-Oriented Database Systems', '31', 'Y', 'Y', 'Y', 'Y'), 'Per', 'A005', 'A006');

1 row created.

SQL> INSERT INTO Papers VALUES
  2  ('5', PAPER ('Relational completeness of database sublanguages', '22', 'Y', 'Y', 'Y', 'Y'), 'DBS', 'A002', 'A000');

1 row created.

SQL> INSERT INTO Papers VALUES
  2  ('6', PAPER ('Modelling uncertainty', '32', 'Y', 'Y', 'Y', 'Y'), 'AI', 'A008', 'A000');

1 row created.

SQL> INSERT INTO Papers VALUES
  2  ('7', PAPER ('A framework for schema updates in an OO Database System', '27', 'Y', 'Y', 'Y', 'Y'), 'DBS', 'A007', 'A000');

1 row created.

SQL> INSERT INTO Papers VALUES
  2  ('8', PAPER ('Constraint-based reasoning', '43', 'Y', 'Y', 'Y', 'Y'), 'DBS', 'A009', 'A000');

1 row created.

SQL> INSERT INTO Papers VALUES
  2  ('9', PAPER ('Relational database: a practical foundation for productivity', '36', 'Y', 'Y', 'Y', 'Y'), 'DBS', 'A002', 'A000');

1 row created.
```

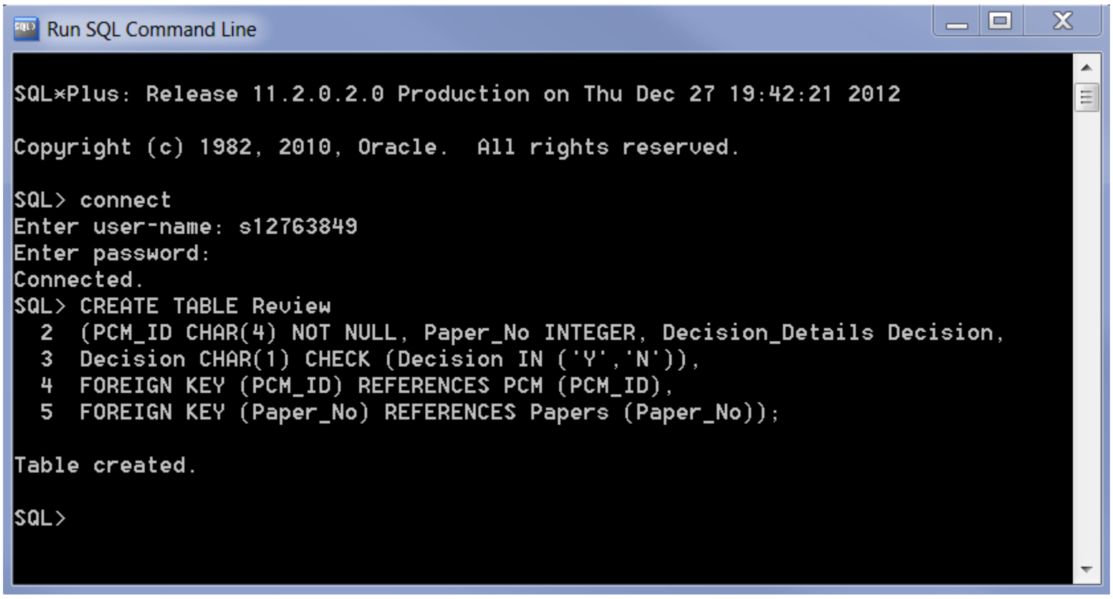


EDIT	PAPER_NO	PAPER_DETAILS	TOPIC_ID	AUTHOR_ID1	AUTHOR_ID2
✎	1	[datatype]	DBS	A001	A000
✎	2	[datatype]	DBS	A002	A000
✎	3	[datatype]	Ind	A003	A004
✎	4	[datatype]	Per	A005	A006
✎	5	[datatype]	DBS	A002	A000
✎	6	[datatype]	AI	A008	A000
✎	7	[datatype]	DBS	A007	A000
✎	8	[datatype]	DBS	A009	A000
✎	9	[datatype]	DBS	A002	A000

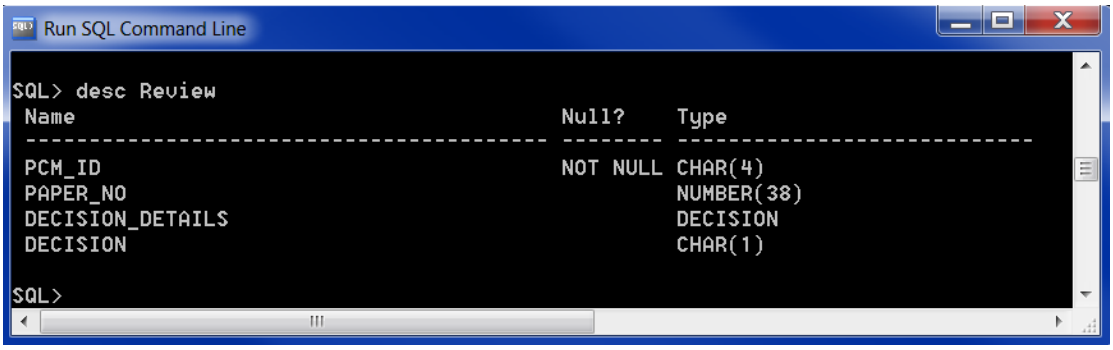
row(s) 1 - 9 of 9

Review

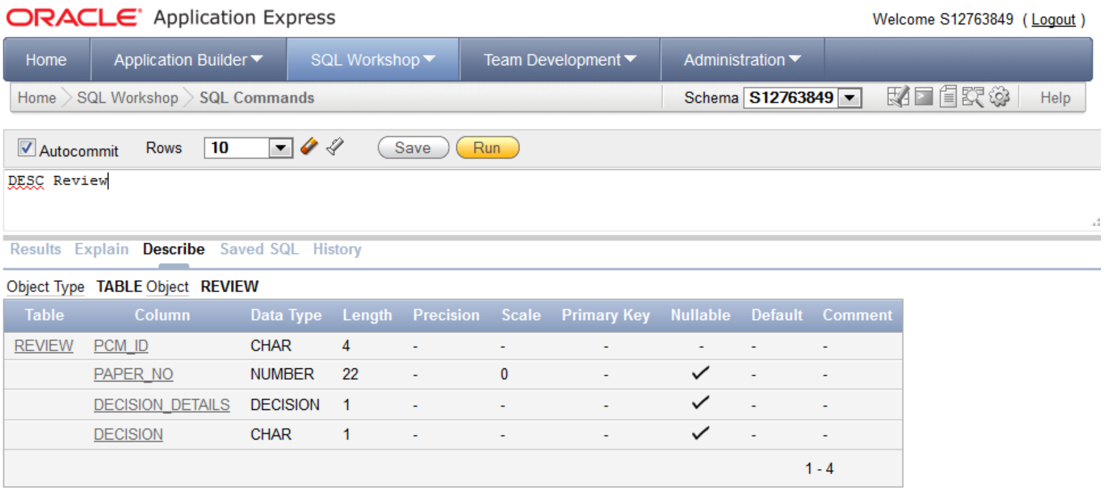
```
CREATE TABLE Review
(PCM_ID CHAR (4) NOT NULL,
Paper_No INTEGER,
Decision_Details Decision,
Decision CHAR (1) CHECK (Decision IN ('Y','N')),
FOREIGN KEY (PCM_ID) REFERENCES PCM (PCM_ID),
FOREIGN KEY (Paper_No) REFERENCES Papers (Paper_No));
```



```
SQL*Plus: Release 11.2.0.2.0 Production on Thu Dec 27 19:42:21 2012

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

SQL> connect
Enter user-name: s12763849
Enter password:
Connected.
SQL> CREATE TABLE Review
  2  (PCM_ID CHAR(4) NOT NULL, Paper_No INTEGER, Decision_Details Decision,
  3  Decision CHAR(1) CHECK (Decision IN ('Y','N')),
  4  FOREIGN KEY (PCM_ID) REFERENCES PCM (PCM_ID),
  5  FOREIGN KEY (Paper_No) REFERENCES Papers (Paper_No));

Table created.

SQL>
```



```
SQL> desc Review
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 PCM_ID                                    NOT NULL CHAR(4)
 PAPER_NO                                           NUMBER(38)
 DECISION_DETAILS                                   DECISION
 DECISION                                           CHAR(1)

SQL>
```



```
DESC Review
```

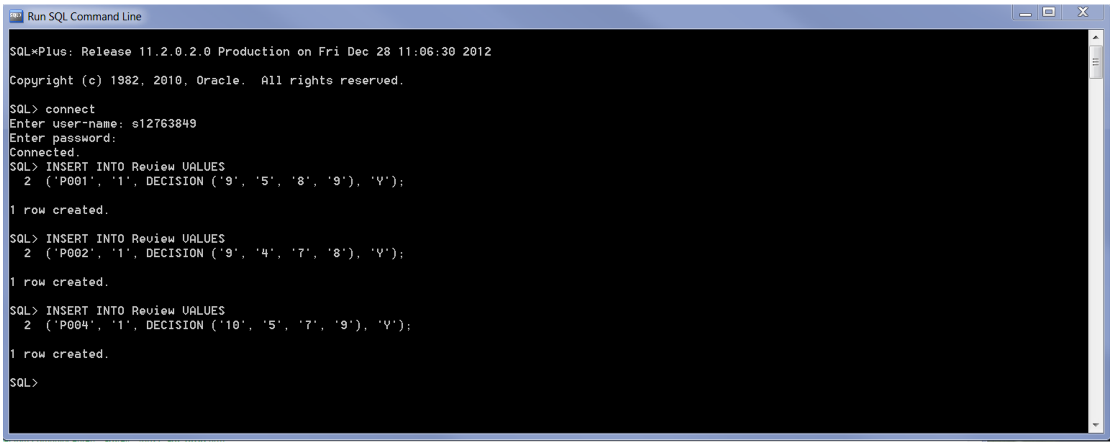
Object Type TABLE Object REVIEW

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
REVIEW	PCM_ID	CHAR	4	-	-	-	-	-	-
	PAPER_NO	NUMBER	22	-	0	-	✓	-	-
	DECISION_DETAILS	DECISION	1	-	-	-	✓	-	-
	DECISION	CHAR	1	-	-	-	✓	-	-

1 - 4

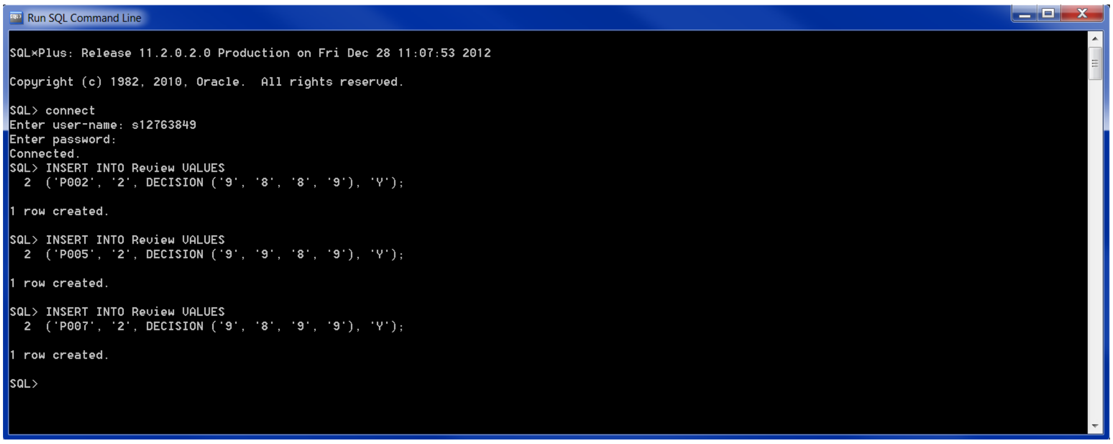
Populated Table with SQL Commands

```
INSERT INTO Review VALUES
('P001', '1', DECISION ('9', '5', '8', '9'), 'Y');
INSERT INTO Review VALUES
('P002', '1', DECISION ('9', '4', '7', '8'), 'Y');
INSERT INTO Review VALUES
('P004', '1', DECISION ('10', '5', '7', '9'), 'Y');
```



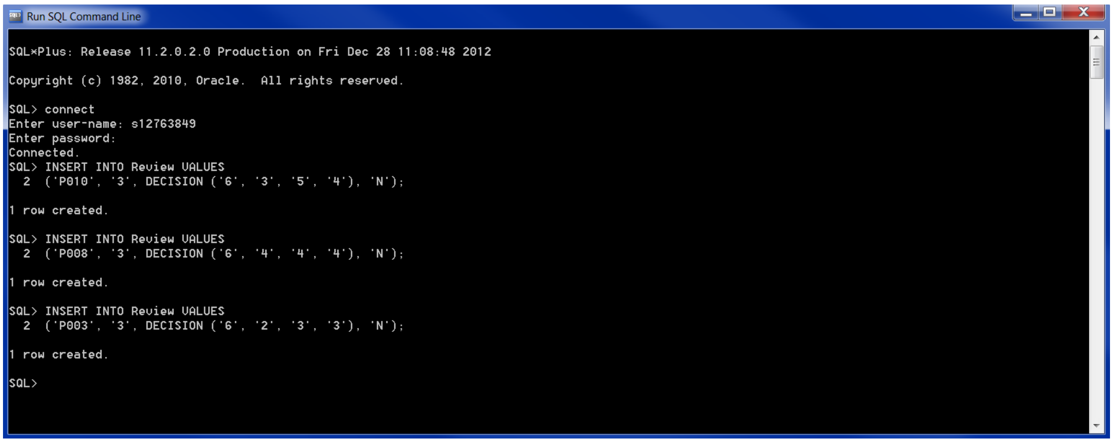
```
SQL*Plus: Release 11.2.0.2.0 Production on Fri Dec 28 11:06:30 2012

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

SQL> connect
Enter user-name: s12763849
Enter password:
Connected.
SQL> INSERT INTO Review VALUES
  2  ('P001', '1', DECISION ('9', '5', '8', '9'), 'Y');

1 row created.

SQL> INSERT INTO Review VALUES
  2  ('P002', '1', DECISION ('9', '4', '7', '8'), 'Y');

1 row created.

SQL> INSERT INTO Review VALUES
  2  ('P004', '1', DECISION ('10', '5', '7', '9'), 'Y');

1 row created.

SQL>
```

```
INSERT INTO Review VALUES
('P002', '2', DECISION ('9', '8', '8', '9'), 'Y');
INSERT INTO Review VALUES
('P005', '2', DECISION ('9', '9', '8', '9'), 'Y');
INSERT INTO Review VALUES
('P007', '2', DECISION ('9', '8', '9', '9'), 'Y');
```



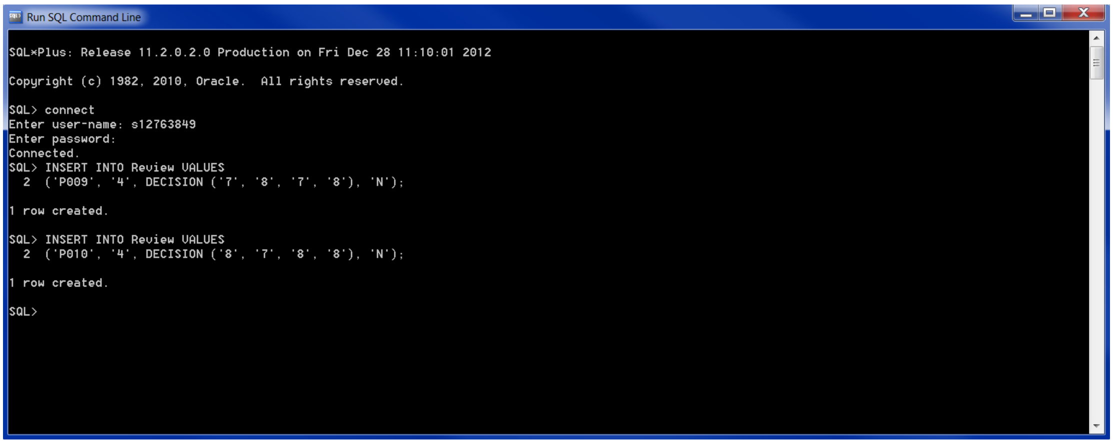
```
SQL*Plus: Release 11.2.0.2.0 Production on Fri Dec 28 11:07:53 2012

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

SQL> connect
Enter user-name: s12763849
Enter password:
Connected.
SQL> INSERT INTO Review VALUES
  2  ('P002', '2', DECISION ('9', '8', '8', '9'), 'Y');

1 row created.

SQL> INSERT INTO Review VALUES
  2  ('P005', '2', DECISION ('9', '9', '8', '9'), 'Y');

1 row created.

SQL> INSERT INTO Review VALUES
  2  ('P007', '2', DECISION ('9', '8', '9', '9'), 'Y');

1 row created.

SQL>
```

```
INSERT INTO Review VALUES
('P010', '3', DECISION ('6', '3', '5', '4'), 'N');
INSERT INTO Review VALUES
('P008', '3', DECISION ('6', '4', '4', '4'), 'N');
INSERT INTO Review VALUES
('P003', '3', DECISION ('6', '2', '3', '3'), 'N');
```



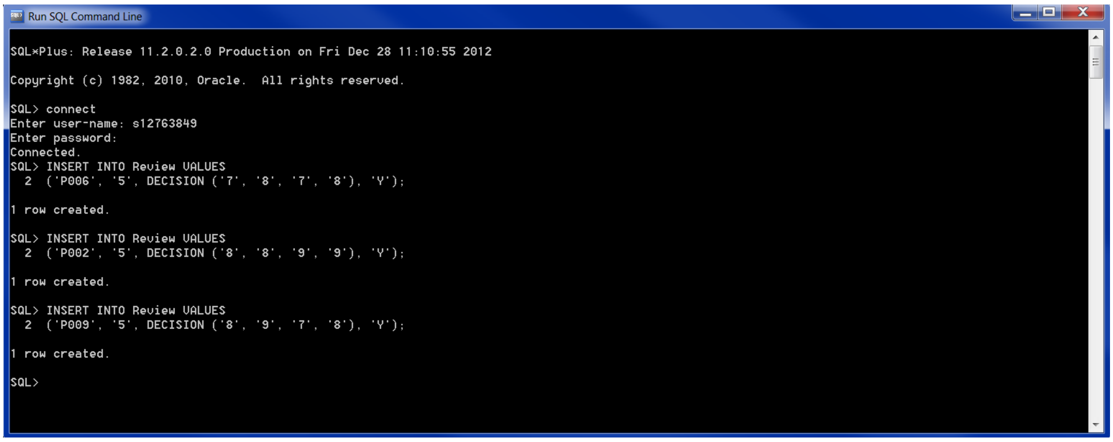
```
SQL*Plus: Release 11.2.0.2.0 Production on Fri Dec 28 11:08:48 2012

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

SQL> connect
Enter user-name: s12763849
Enter password:
Connected.
SQL> INSERT INTO Review VALUES
  2  ('P010', '3', DECISION ('6', '3', '5', '4'), 'N');

1 row created.

SQL> INSERT INTO Review VALUES
  2  ('P008', '3', DECISION ('6', '4', '4', '4'), 'N');

1 row created.

SQL> INSERT INTO Review VALUES
  2  ('P003', '3', DECISION ('6', '2', '3', '3'), 'N');

1 row created.

SQL>
```

```
INSERT INTO Review VALUES
('P009', '4', DECISION ('7', '8', '7', '8'), 'N');
INSERT INTO Review VALUES
('P010', '4', DECISION ('8', '7', '8', '8'), 'N');
```



```
SQL*Plus: Release 11.2.0.2.0 Production on Fri Dec 28 11:10:01 2012

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

SQL> connect
Enter user-name: s12763849
Enter password:
Connected.
SQL> INSERT INTO Review VALUES
  2  ('P009', '4', DECISION ('7', '8', '7', '8'), 'N');

1 row created.

SQL> INSERT INTO Review VALUES
  2  ('P010', '4', DECISION ('8', '7', '8', '8'), 'N');

1 row created.

SQL>
```

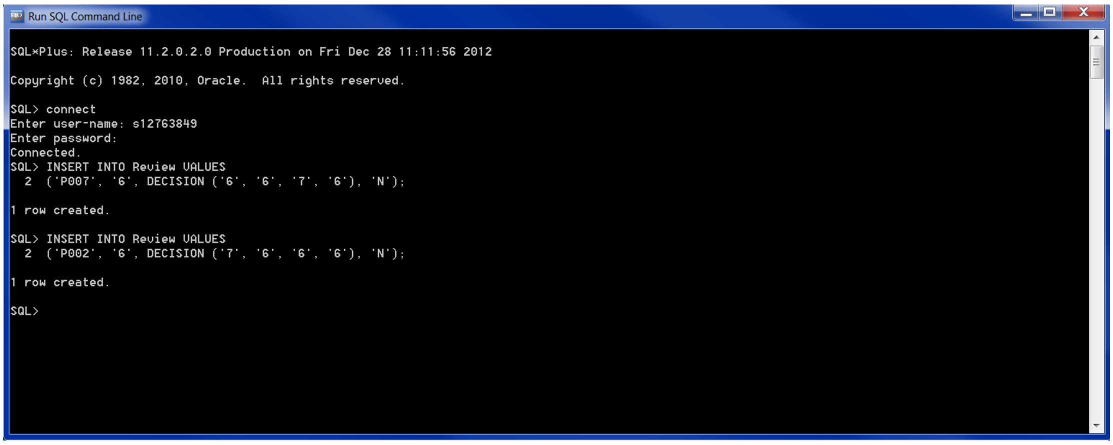
```
INSERT INTO Review VALUES
('P006', '5', DECISION ('7', '8', '7', '8'), 'Y');
INSERT INTO Review VALUES
('P002', '5', DECISION ('8', '8', '9', '9'), 'Y');
INSERT INTO Review VALUES
('P009', '5', DECISION ('8', '9', '7', '8'), 'Y');
```



```
SQL*Plus: Release 11.2.0.2.0 Production on Fri Dec 28 11:10:55 2012

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

SQL> connect
Enter user-name: s12763849
Enter password:
Connected.
SQL> INSERT INTO Review VALUES
  2  ('P006', '5', DECISION ('7', '8', '7', '8'), 'Y');

1 row created.

SQL> INSERT INTO Review VALUES
  2  ('P002', '5', DECISION ('8', '8', '9', '9'), 'Y');

1 row created.

SQL> INSERT INTO Review VALUES
  2  ('P009', '5', DECISION ('8', '9', '7', '8'), 'Y');

1 row created.

SQL>
```

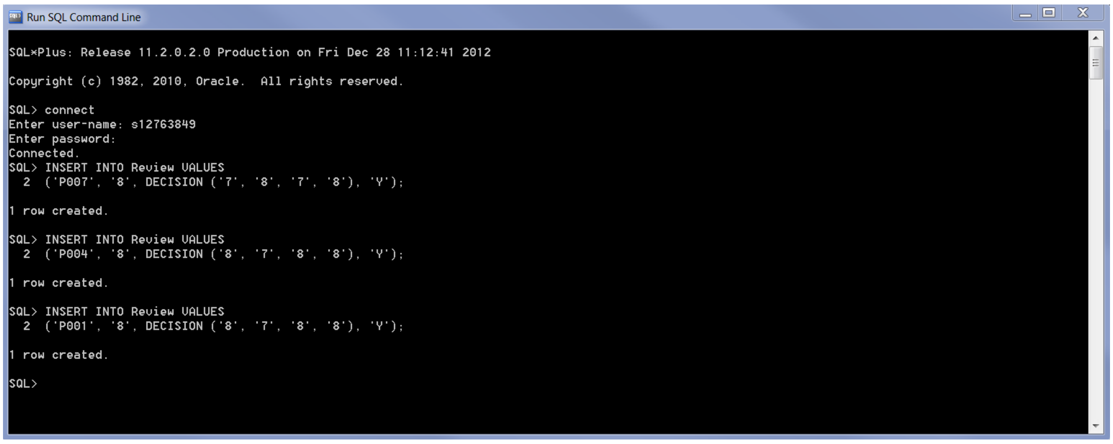
```
INSERT INTO Review VALUES
('P007', '6', DECISION ('6', '6', '7', '6'), 'N');
INSERT INTO Review VALUES
```

```
('P002', '6', DECISION ('7', '6', '6', '6'), 'N');
```



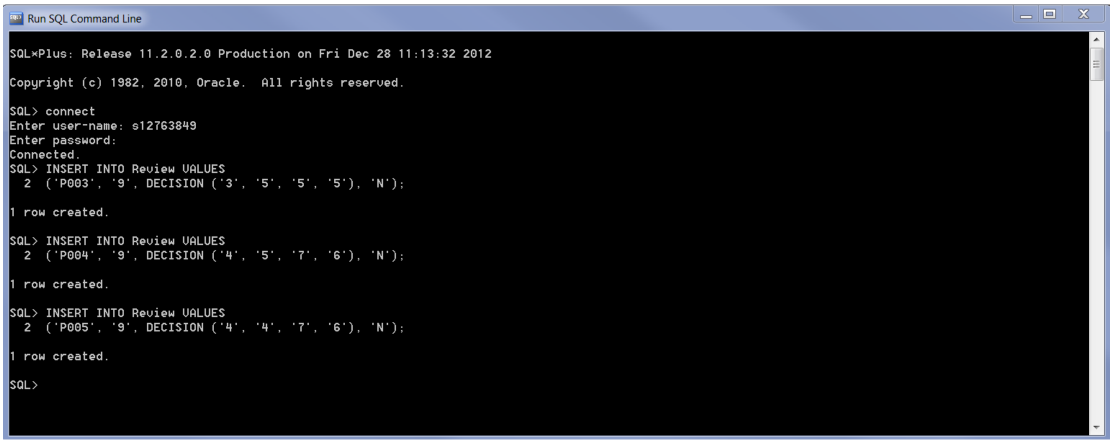
```
SQL*Plus: Release 11.2.0.2.0 Production on Fri Dec 28 11:11:56 2012

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

SQL> connect
Enter user-name: s12763849
Enter password:
Connected.
SQL> INSERT INTO Review VALUES
  2  ('P007', '6', DECISION ('6', '6', '7', '6'), 'N');

1 row created.

SQL> INSERT INTO Review VALUES
  2  ('P002', '6', DECISION ('7', '6', '6', '6'), 'N');

1 row created.

SQL>
```

```
INSERT INTO Review VALUES
('P007', '8', DECISION ('7', '8', '7', '8'), 'Y');
INSERT INTO Review VALUES
('P004', '8', DECISION ('8', '7', '8', '8'), 'Y');
INSERT INTO Review VALUES
('P001', '8', DECISION ('8', '7', '8', '8'), 'Y');
```

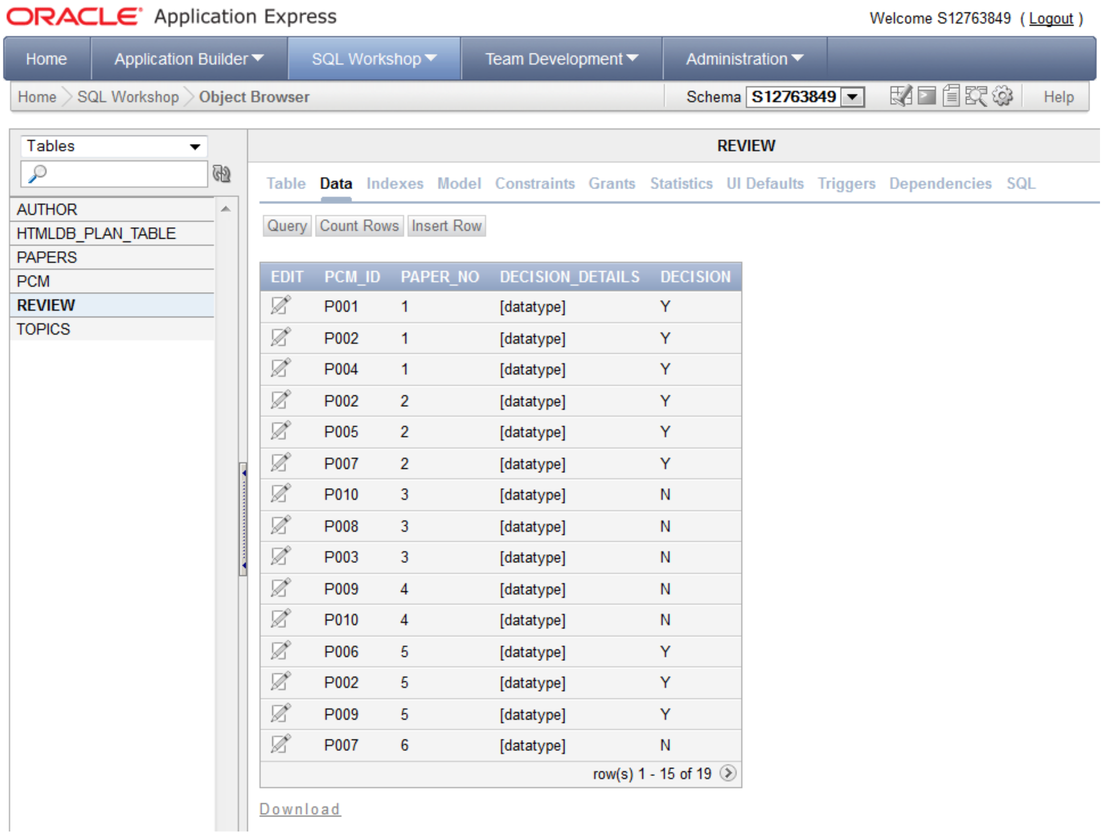


```
SQL*Plus: Release 11.2.0.2.0 Production on Fri Dec 28 11:12:41 2012

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

SQL> connect
Enter user-name: s12763849
Enter password:
Connected.
SQL> INSERT INTO Review VALUES
  2  ('P007', '8', DECISION ('7', '8', '7', '8'), 'Y');

1 row created.

SQL> INSERT INTO Review VALUES
  2  ('P004', '8', DECISION ('8', '7', '8', '8'), 'Y');

1 row created.

SQL> INSERT INTO Review VALUES
  2  ('P001', '8', DECISION ('8', '7', '8', '8'), 'Y');

1 row created.

SQL>
```

```
INSERT INTO Review VALUES
('P003', '9', DECISION ('3', '5', '5', '5'), 'N');
INSERT INTO Review VALUES
('P004', '9', DECISION ('4', '5', '7', '6'), 'N');
INSERT INTO Review VALUES
('P005', '9', DECISION ('4', '4', '7', '6'), 'N');
```

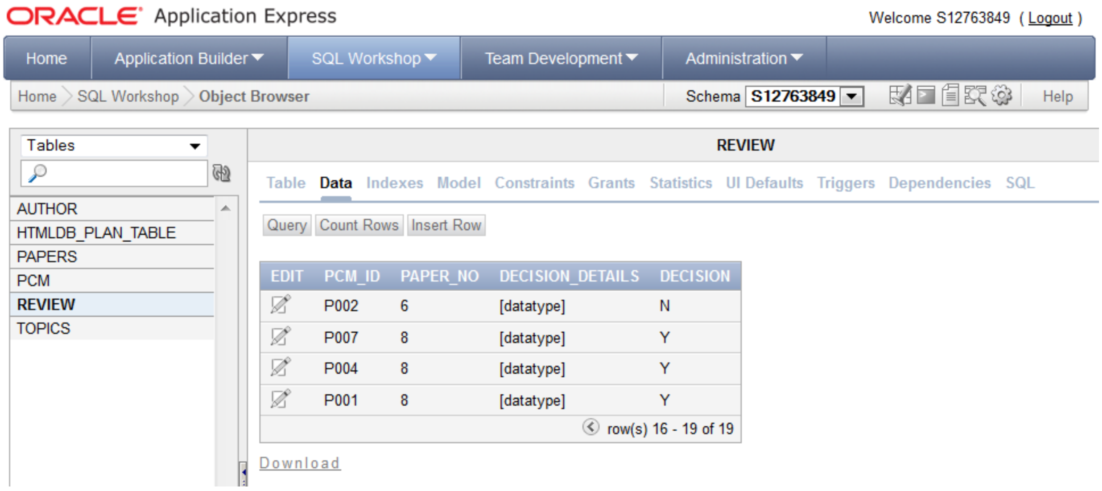


```
SQL*Plus: Release 11.2.0.2.0 Production on Fri Dec 28 11:13:32 2012

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

SQL> connect
Enter user-name: s12763849
Enter password:
Connected.
SQL> INSERT INTO Review VALUES
  2  ('P003', '9', DECISION ('3', '5', '5', '5'), 'N');

1 row created.

SQL> INSERT INTO Review VALUES
  2  ('P004', '9', DECISION ('4', '5', '7', '6'), 'N');

1 row created.

SQL> INSERT INTO Review VALUES
  2  ('P005', '9', DECISION ('4', '4', '7', '6'), 'N');

1 row created.

SQL>
```



EDIT	PCM_ID	PAPER_NO	DECISION_DETAILS	DECISION
	P001	1	[datatype]	Y
	P002	1	[datatype]	Y
	P004	1	[datatype]	Y
	P002	2	[datatype]	Y
	P005	2	[datatype]	Y
	P007	2	[datatype]	Y
	P010	3	[datatype]	N
	P008	3	[datatype]	N
	P003	3	[datatype]	N
	P009	4	[datatype]	N
	P010	4	[datatype]	N
	P006	5	[datatype]	Y
	P002	5	[datatype]	Y
	P009	5	[datatype]	Y
	P007	6	[datatype]	N

row(s) 1 - 15 of 19



EDIT	PCM_ID	PAPER_NO	DECISION_DETAILS	DECISION
	P002	6	[datatype]	N
	P007	8	[datatype]	Y
	P004	8	[datatype]	Y
	P001	8	[datatype]	Y

row(s) 16 - 19 of 19

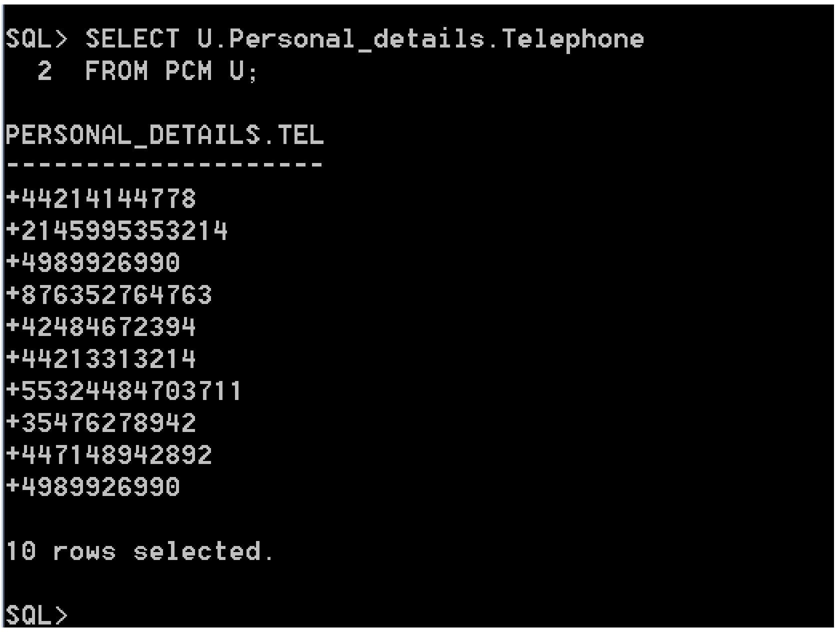
Task 2

*Run some sample SQL queries that explicitly utilize the new OO features. Use your imagination or modify the queries from assessment A. Turn the **timer** on and time the performance of these queries.*

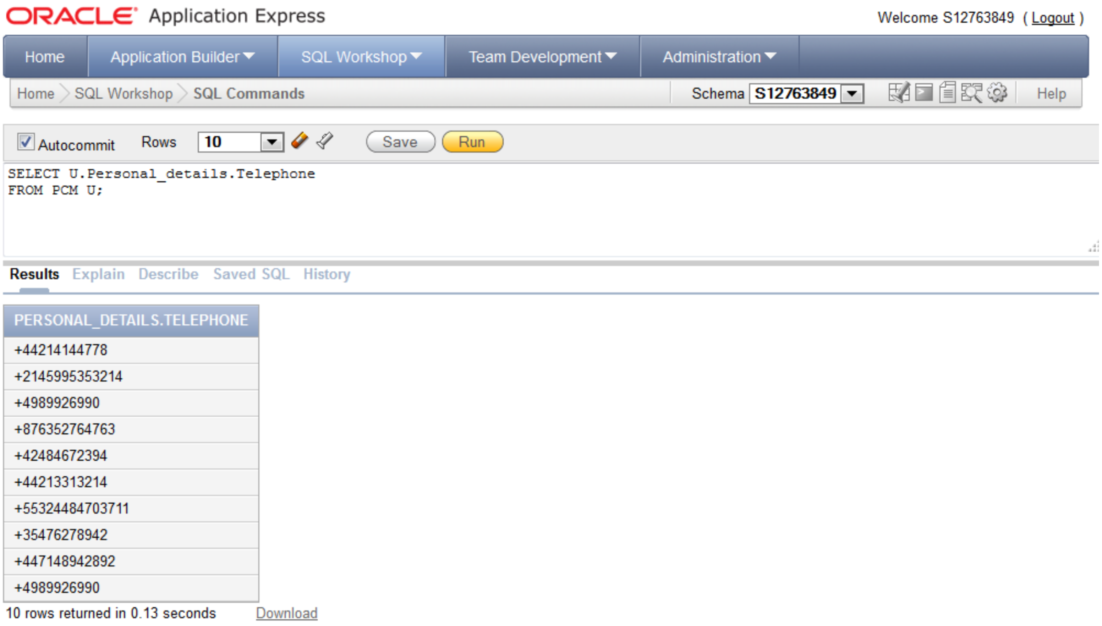
SQL Queries using Object Orientated Features

Show phone numbers from object person nested in PCM table

```
SELECT U.Personal_details.Telephone
FROM PCM U;
```



```
SQL> SELECT U.Personal_details.Telephone
  2  FROM PCM U;

PERSONAL_DETAILS.TEL
--------------------
+44214144778
+2145995353214
+4989926990
+876352764763
+42484672394
+44213313214
+55324484703711
+35476278942
+447148942892
+4989926990

10 rows selected.

SQL>
```



Schema S12763849

☑ Autocommit Rows 10

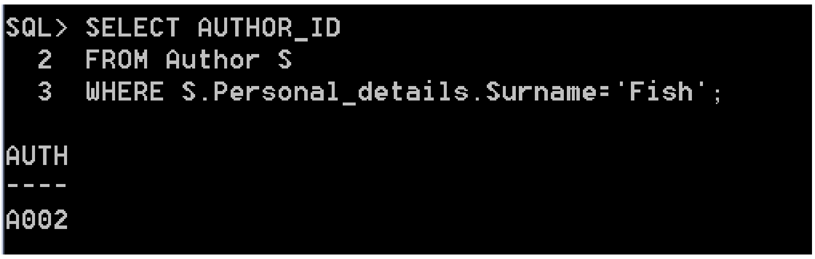
```
SELECT U.Personal_details.Telephone
FROM PCM U;
```

PERSONAL_DETAILS.TELEPHONE
+44214144778
+2145995353214
+4989926990
+876352764763
+42484672394
+44213313214
+55324484703711
+35476278942
+447148942892
+4989926990

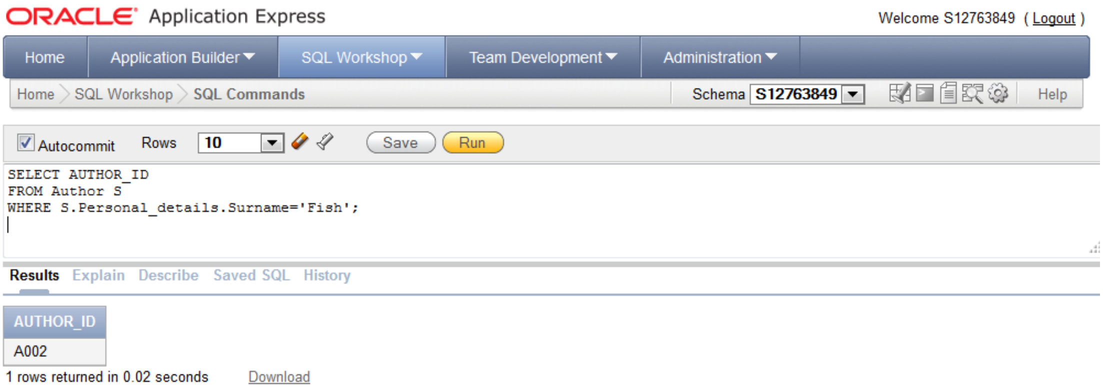
10 rows returned in 0.13 seconds

Show author ID for author with surname Fish

```
SELECT AUTHOR_ID
FROM Author S
WHERE S.Personal_details.Surname='Fish';
```



```
SQL> SELECT AUTHOR_ID
  2  FROM Author S
  3  WHERE S.Personal_details.Surname='Fish';

AUTH
----
A002
```



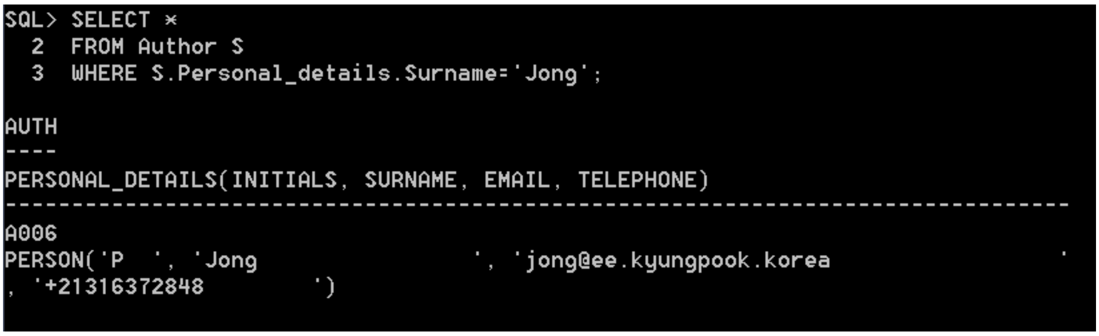
```
SELECT AUTHOR_ID
FROM Author S
WHERE S.Personal_details.Surname='Fish';
```

AUTHOR_ID
A002

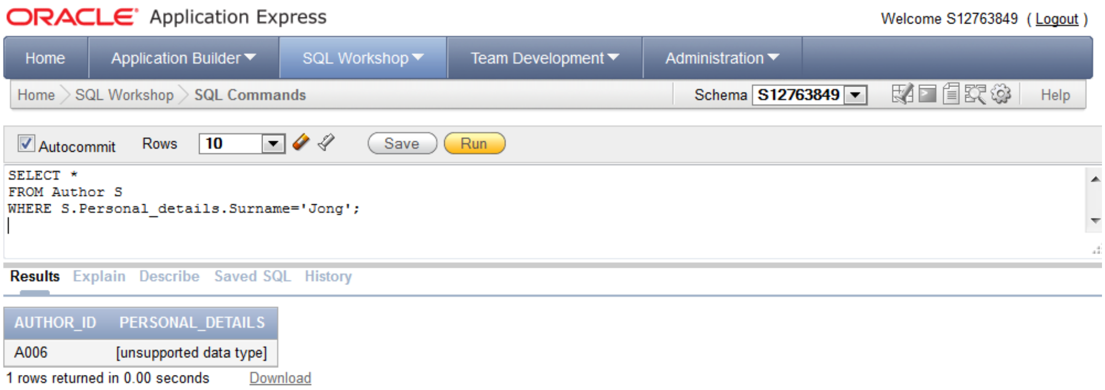
1 rows returned in 0.02 seconds

Show all details for author with surname Jong

```
SELECT *
FROM Author S
WHERE S.Personal_details.Surname='Jong';
```



```
SQL> SELECT *
  2  FROM Author S
  3  WHERE S.Personal_details.Surname='Jong';

AUTH
----
PERSONAL_DETAILS(INITIALS, SURNAME, EMAIL, TELEPHONE)
--------------------------------------------------------------------------------
A006
PERSON('P  ', 'Jong                    ', 'jong@ee.kyungpook.korea                 '
, '+21316372848       ')
```



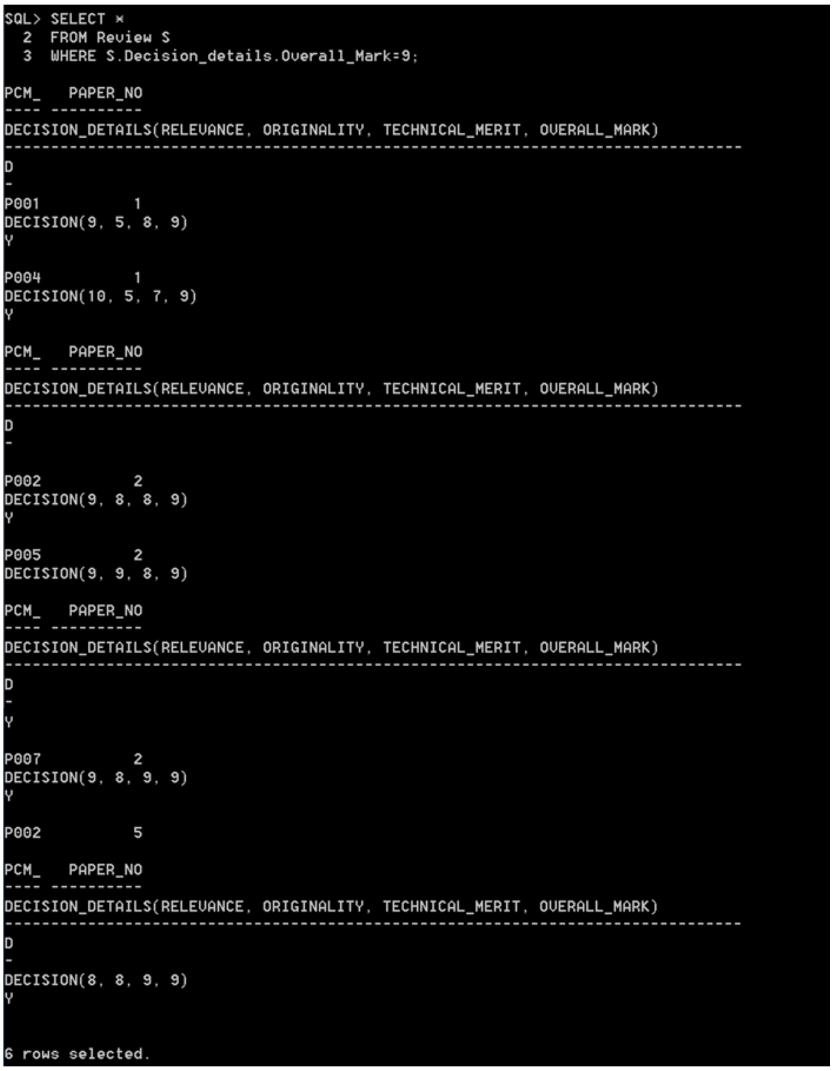
```
SELECT *
FROM Author S
WHERE S.Personal_details.Surname='Jong';
```

AUTHOR_ID	PERSONAL_DETAILS
A006	[unsupported data type]

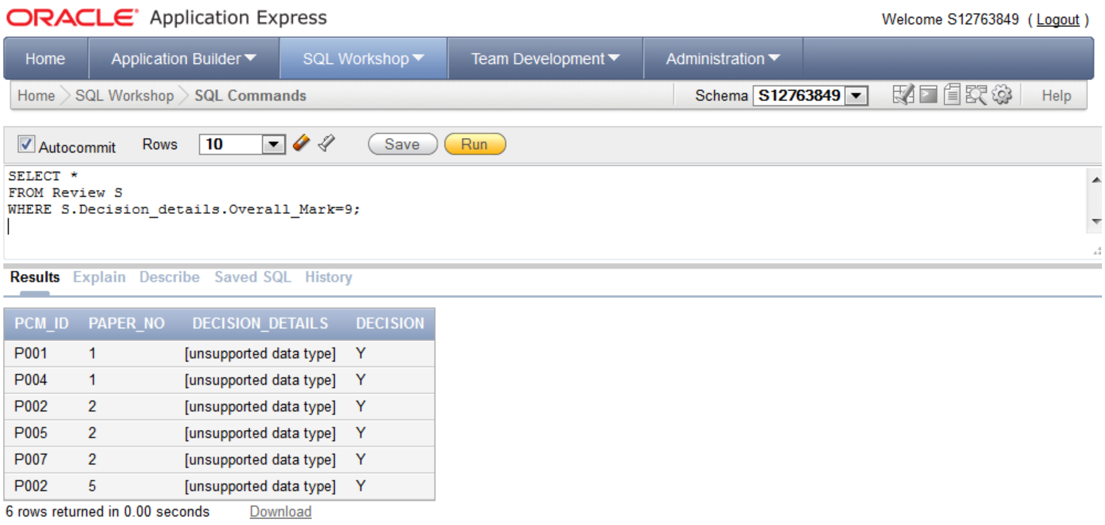
1 rows returned in 0.00 seconds

Show all details from review where overall mark is 9

```
SELECT *
FROM Review S
WHERE S.Decision_details.Overall_Mark=9;
```



```
SQL> SELECT *
  2  FROM Review S
  3  WHERE S.Decision_details.Overall_Mark=9;

PCM_   PAPER_NO
---- ----------
DECISION_DETAILS(RELEVANCE, ORIGINALITY, TECHNICAL_MERIT, OVERALL_MARK)
--------------------------------------------------------------------------------
D
-
P001          1
DECISION(9, 5, 8, 9)
Y

P004          1
DECISION(10, 5, 7, 9)
Y

PCM_   PAPER_NO
---- ----------
DECISION_DETAILS(RELEVANCE, ORIGINALITY, TECHNICAL_MERIT, OVERALL_MARK)
--------------------------------------------------------------------------------
D
-

P002          2
DECISION(9, 8, 8, 9)
Y

P005          2
DECISION(9, 9, 8, 9)

PCM_   PAPER_NO
---- ----------
DECISION_DETAILS(RELEVANCE, ORIGINALITY, TECHNICAL_MERIT, OVERALL_MARK)
--------------------------------------------------------------------------------
D
-
Y

P007          2
DECISION(9, 8, 9, 9)
Y

P002          5

PCM_   PAPER_NO
---- ----------
DECISION_DETAILS(RELEVANCE, ORIGINALITY, TECHNICAL_MERIT, OVERALL_MARK)
--------------------------------------------------------------------------------
D
-
DECISION(8, 8, 9, 9)
Y


6 rows selected.
```



☑ Autocommit Rows 10

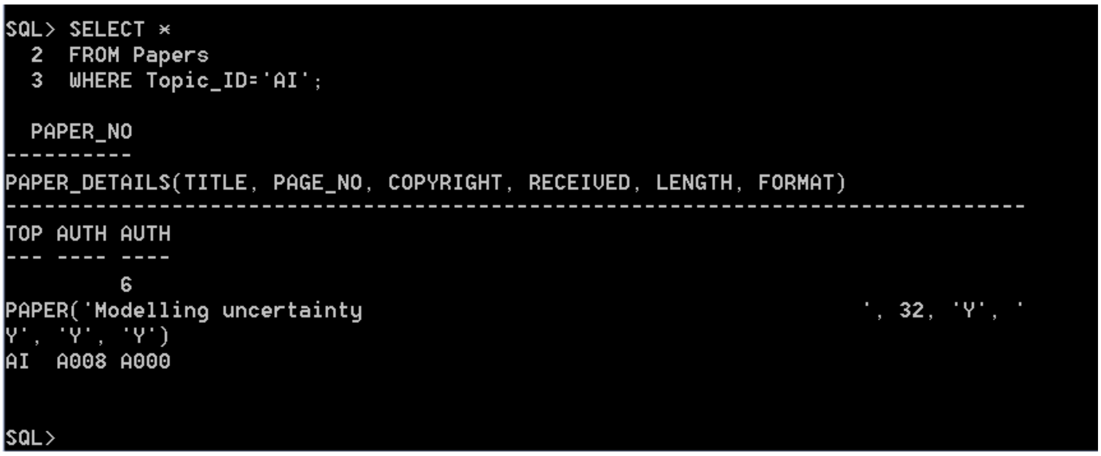
```
SELECT *
FROM Review S
WHERE S.Decision_details.Overall_Mark=9;
```

PCM_ID	PAPER_NO	DECISION_DETAILS	DECISION
P001	1	[unsupported data type]	Y
P004	1	[unsupported data type]	Y
P002	2	[unsupported data type]	Y
P005	2	[unsupported data type]	Y
P007	2	[unsupported data type]	Y
P002	5	[unsupported data type]	Y

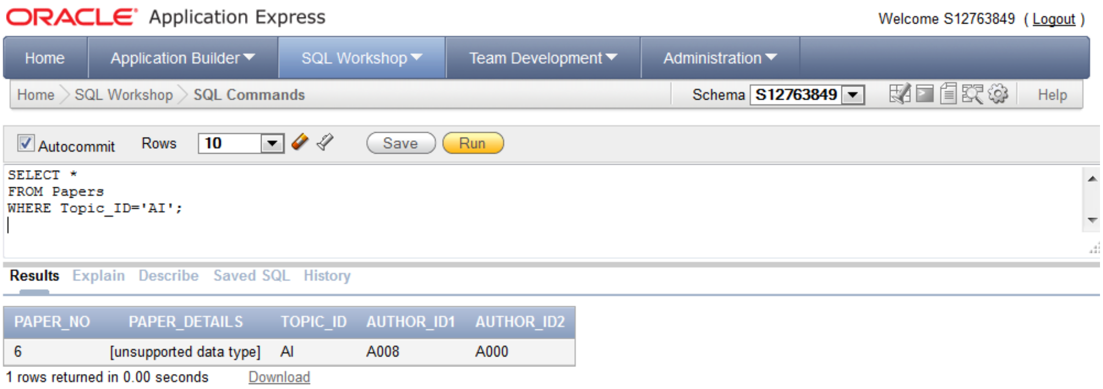
6 rows returned in 0.00 seconds

Show all papers where topic ID is AI

```
SELECT *
FROM Papers
WHERE Topic_ID='AI';
```



```
SQL> SELECT *
  2  FROM Papers
  3  WHERE Topic_ID='AI';

  PAPER_NO
----------
PAPER_DETAILS(TITLE, PAGE_NO, COPYRIGHT, RECEIVED, LENGTH, FORMAT)
--------------------------------------------------------------------------------
TOP AUTH AUTH
--- ---- ----
         6
PAPER('Modelling uncertainty                                        ', 32, 'Y', '
Y', 'Y', 'Y')
AI  A008 A000


SQL>
```



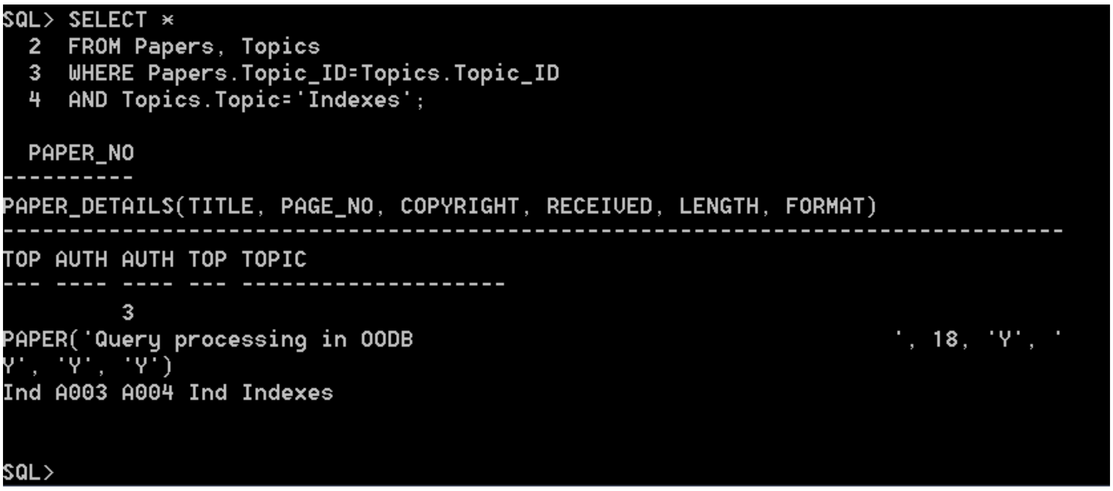
```
SELECT *
FROM Papers
WHERE Topic_ID='AI';
```

PAPER_NO	PAPER_DETAILS	TOPIC_ID	AUTHOR_ID1	AUTHOR_ID2
6	[unsupported data type]	AI	A008	A000

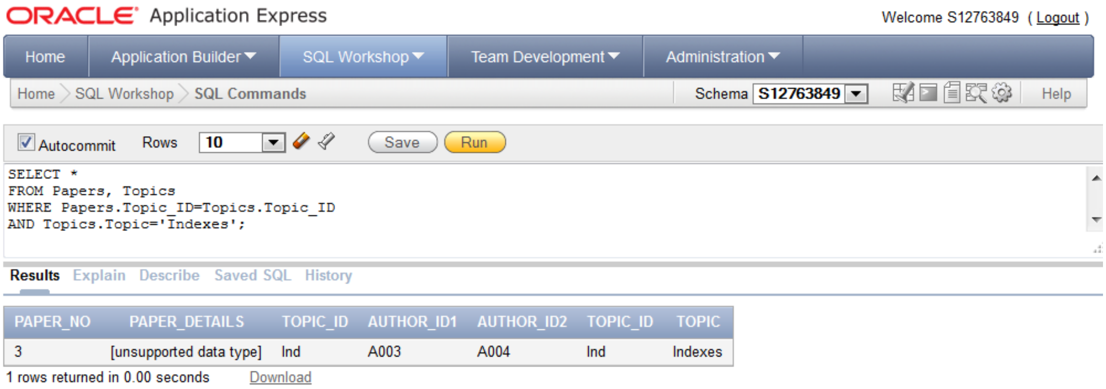
1 rows returned in 0.00 seconds   Download

Show all papers where topic is indexes

```
SELECT *
FROM Papers, Topics
WHERE Papers.Topic_ID=Topics.Topic_ID
AND Topics.Topic='Indexes';
```



```
SQL> SELECT *
  2  FROM Papers, Topics
  3  WHERE Papers.Topic_ID=Topics.Topic_ID
  4  AND Topics.Topic='Indexes';

  PAPER_NO
----------
PAPER_DETAILS(TITLE, PAGE_NO, COPYRIGHT, RECEIVED, LENGTH, FORMAT)
--------------------------------------------------------------------------------
TOP AUTH AUTH TOP TOPIC
--- ---- ---- --- --------------------
         3
PAPER('Query processing in OODB                                    ', 18, 'Y', '
Y', 'Y', 'Y')
Ind A003 A004 Ind Indexes


SQL>
```



ORACLE Application Express
Welcome S12763849 (Logout)
Home
Application Builder
SQL Workshop
Team Development
Administration
Home > SQL Workshop > SQL Commands
Schema S12763849
Help
☑ Autocommit
Rows 10
Save
Run
Results Explain Describe Saved SQL History

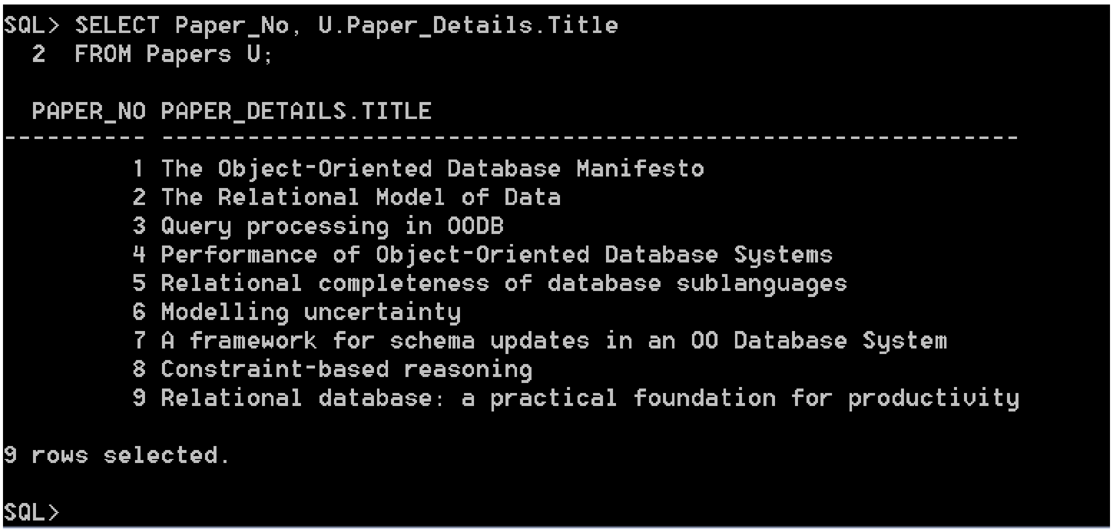
```
SELECT *
FROM Papers, Topics
WHERE Papers.Topic_ID=Topics.Topic_ID
AND Topics.Topic='Indexes';
```

PAPER_NO	PAPER_DETAILS	TOPIC_ID	AUTHOR_ID1	AUTHOR_ID2	TOPIC_ID	TOPIC
3	[unsupported data type]	Ind	A003	A004	Ind	Indexes

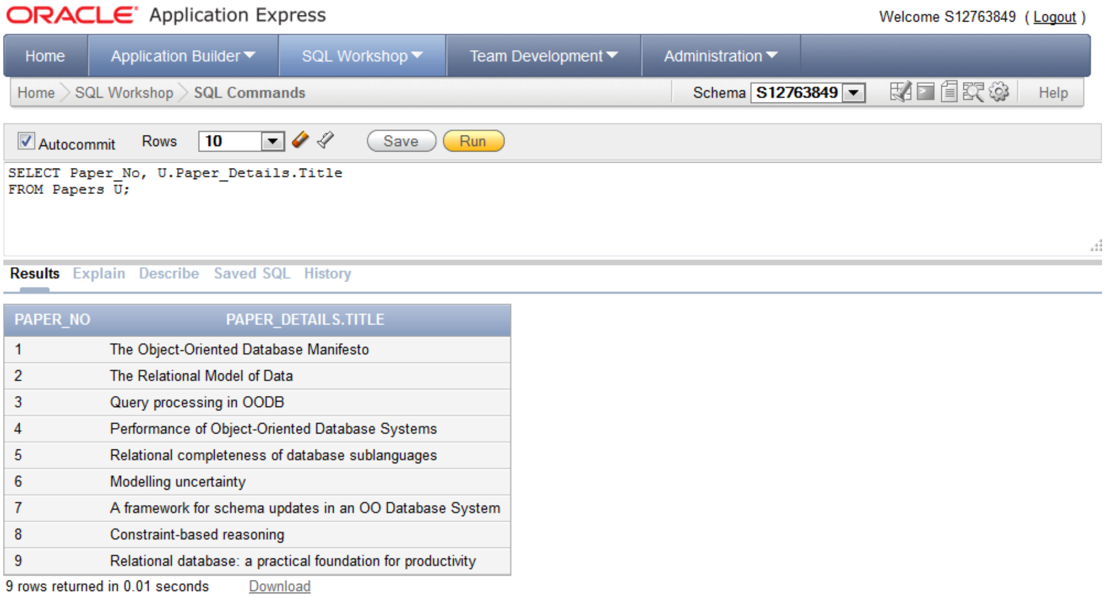
1 rows returned in 0.00 seconds Download

Show all paper numbers and titles

```
SELECT Paper_No, U.Paper_Details.Title
FROM Papers U;
```



```
SQL> SELECT Paper_No, U.Paper_Details.Title
  2  FROM Papers U;

  PAPER_NO PAPER_DETAILS.TITLE
---------- ------------------------------------------------------------
         1 The Object-Oriented Database Manifesto
         2 The Relational Model of Data
         3 Query processing in OODB
         4 Performance of Object-Oriented Database Systems
         5 Relational completeness of database sublanguages
         6 Modelling uncertainty
         7 A framework for schema updates in an OO Database System
         8 Constraint-based reasoning
         9 Relational database: a practical foundation for productivity

9 rows selected.

SQL>
```



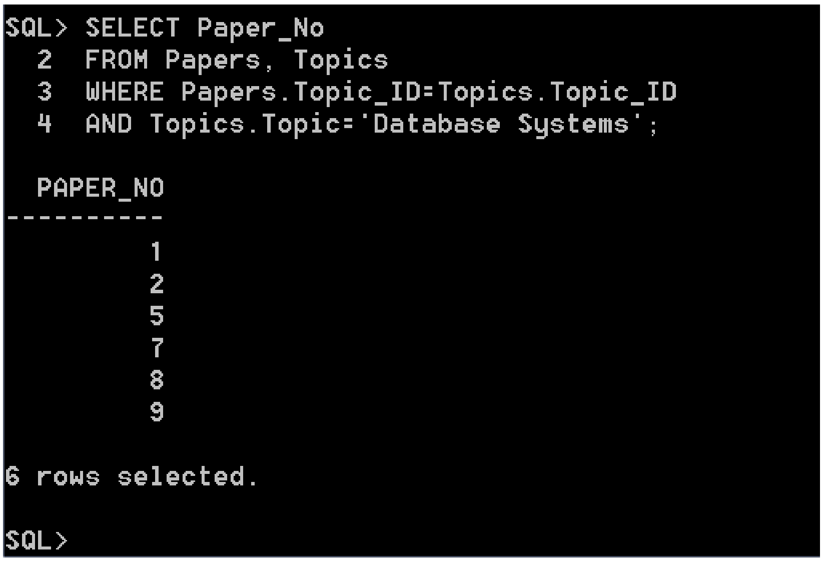
```
SELECT Paper_No, U.Paper_Details.Title
FROM Papers U;
```

PAPER_NO	PAPER_DETAILS.TITLE
1	The Object-Oriented Database Manifesto
2	The Relational Model of Data
3	Query processing in OODB
4	Performance of Object-Oriented Database Systems
5	Relational completeness of database sublanguages
6	Modelling uncertainty
7	A framework for schema updates in an OO Database System
8	Constraint-based reasoning
9	Relational database: a practical foundation for productivity

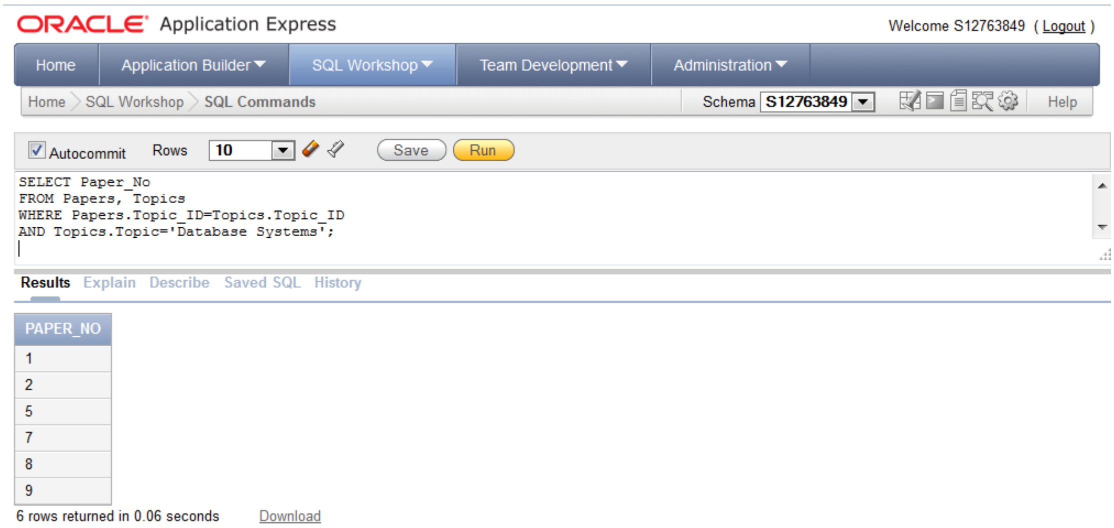
9 rows returned in 0.01 seconds   Download

Show paper number for all papers where topic is Database Systems

```
SELECT Paper_No
FROM Papers, Topics
WHERE Papers.Topic_ID=Topics.Topic_ID
AND Topics.Topic='Database Systems';
```



```
SQL> SELECT Paper_No
  2  FROM Papers, Topics
  3  WHERE Papers.Topic_ID=Topics.Topic_ID
  4  AND Topics.Topic='Database Systems';

  PAPER_NO
----------
         1
         2
         5
         7
         8
         9

6 rows selected.

SQL>
```



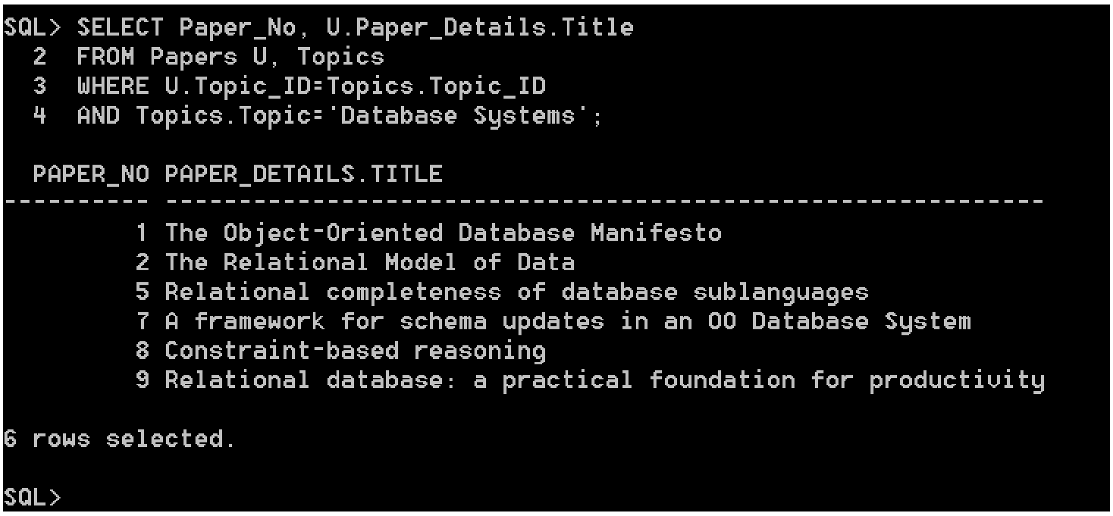
```
SELECT Paper_No
FROM Papers, Topics
WHERE Papers.Topic_ID=Topics.Topic_ID
AND Topics.Topic='Database Systems';
```

PAPER_NO
1
2
5
7
8
9

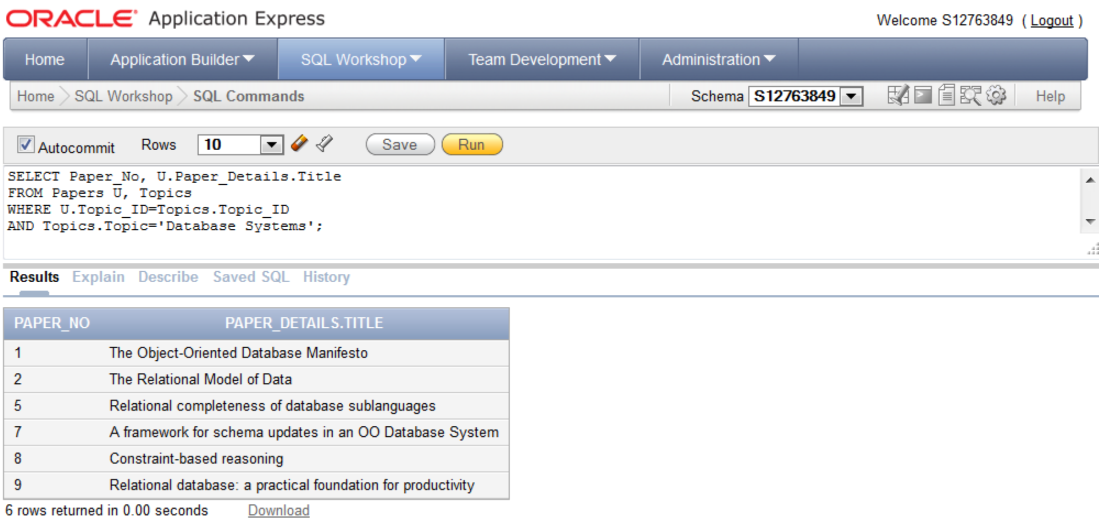
6 rows returned in 0.06 seconds

Show paper numbers and titles for all papers where topic is Database Systems

```
SELECT Paper_No, U.Paper_Details.Title
FROM Papers U, Topics
WHERE U.Topic_ID=Topics.Topic_ID
AND Topics.Topic='Database Systems';
```



```
SQL> SELECT Paper_No, U.Paper_Details.Title
  2  FROM Papers U, Topics
  3  WHERE U.Topic_ID=Topics.Topic_ID
  4  AND Topics.Topic='Database Systems';

  PAPER_NO PAPER_DETAILS.TITLE
---------- ------------------------------------------------------------
         1 The Object-Oriented Database Manifesto
         2 The Relational Model of Data
         5 Relational completeness of database sublanguages
         7 A framework for schema updates in an OO Database System
         8 Constraint-based reasoning
         9 Relational database: a practical foundation for productivity

6 rows selected.

SQL>
```



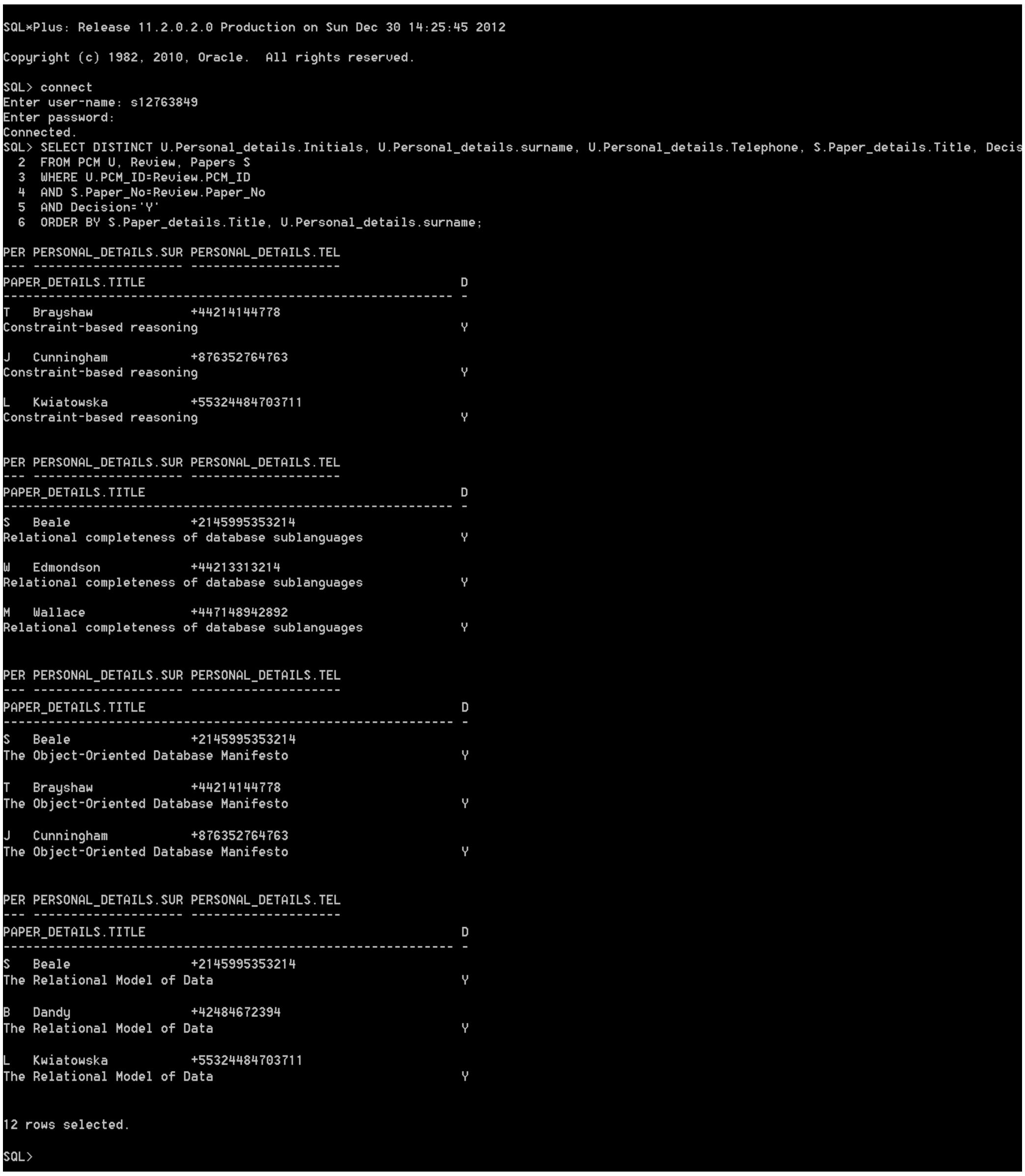
```
SELECT Paper_No, U.Paper_Details.Title
FROM Papers U, Topics
WHERE U.Topic_ID=Topics.Topic_ID
AND Topics.Topic='Database Systems';
```

PAPER_NO	PAPER_DETAILS.TITLE
1	The Object-Oriented Database Manifesto
2	The Relational Model of Data
5	Relational completeness of database sublanguages
7	A framework for schema updates in an OO Database System
8	Constraint-based reasoning
9	Relational database: a practical foundation for productivity

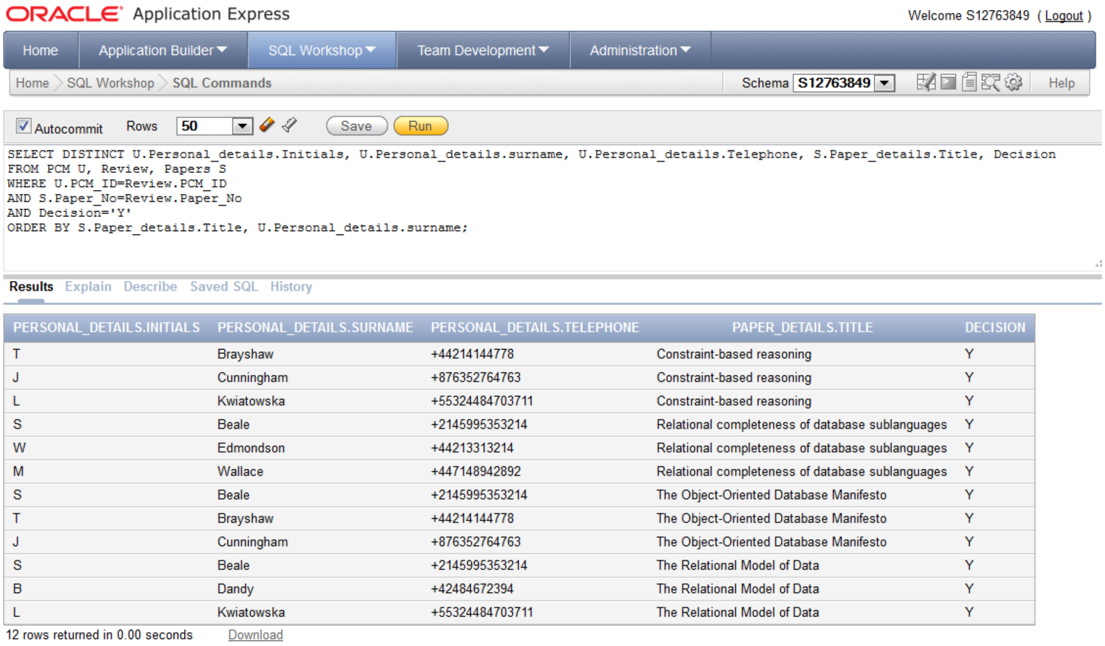
6 rows returned in 0.00 seconds   Download

Show in alphabetical order (by title then surname) the full names of reviewers who've given a positive decision and these papers

```
SELECT DISTINCT U.Personal_details.Initials,
U.Personal_details.surname, U.Personal_details.Telephone,
S.Paper_details.Title, Decision
FROM PCM U, Review, Papers S
WHERE U.PCM_ID=Review.PCM_ID
AND S.Paper_No=Review.Paper_No
AND Decision='Y'
ORDER BY S.Paper_details.Title, U.Personal_details.surname;
```



```
SQL*Plus: Release 11.2.0.2.0 Production on Sun Dec 30 14:25:45 2012

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

SQL> connect
Enter user-name: s12763849
Enter password:
Connected.
SQL> SELECT DISTINCT U.Personal_details.Initials, U.Personal_details.surname, U.Personal_details.Telephone, S.Paper_details.Title, Decis
  2  FROM PCM U, Review, Papers S
  3  WHERE U.PCM_ID=Review.PCM_ID
  4  AND S.Paper_No=Review.Paper_No
  5  AND Decision='Y'
  6  ORDER BY S.Paper_details.Title, U.Personal_details.surname;

PER PERSONAL_DETAILS.SUR PERSONAL_DETAILS.TEL
--- -------------------- --------------------
PAPER_DETAILS.TITLE                                          D
------------------------------------------------------------ -
T   Brayshaw             +44214144778
Constraint-based reasoning                                   Y

J   Cunningham           +876352764763
Constraint-based reasoning                                   Y

L   Kwiatowska           +55324484703711
Constraint-based reasoning                                   Y


PER PERSONAL_DETAILS.SUR PERSONAL_DETAILS.TEL
--- -------------------- --------------------
PAPER_DETAILS.TITLE                                          D
------------------------------------------------------------ -
S   Beale                +2145995353214
Relational completeness of database sublanguages             Y

W   Edmondson            +44213313214
Relational completeness of database sublanguages             Y

M   Wallace              +447148942892
Relational completeness of database sublanguages             Y


PER PERSONAL_DETAILS.SUR PERSONAL_DETAILS.TEL
--- -------------------- --------------------
PAPER_DETAILS.TITLE                                          D
------------------------------------------------------------ -
S   Beale                +2145995353214
The Object-Oriented Database Manifesto                       Y

T   Brayshaw             +44214144778
The Object-Oriented Database Manifesto                       Y

J   Cunningham           +876352764763
The Object-Oriented Database Manifesto                       Y


PER PERSONAL_DETAILS.SUR PERSONAL_DETAILS.TEL
--- -------------------- --------------------
PAPER_DETAILS.TITLE                                          D
------------------------------------------------------------ -
S   Beale                +2145995353214
The Relational Model of Data                                 Y

B   Dandy                +42484672394
The Relational Model of Data                                 Y

L   Kwiatowska           +55324484703711
The Relational Model of Data                                 Y


12 rows selected.

SQL>
```



```
SELECT DISTINCT U.Personal_details.Initials, U.Personal_details.surname, U.Personal_details.Telephone, S.Paper_details.Title, Decision
FROM PCM U, Review, Papers S
WHERE U.PCM_ID=Review.PCM_ID
AND S.Paper_No=Review.Paper_No
AND Decision='Y'
ORDER BY S.Paper_details.Title, U.Personal_details.surname;
```

PERSONAL_DETAILS.INITIALS	PERSONAL_DETAILS.SURNAME	PERSONAL_DETAILS.TELEPHONE	PAPER_DETAILS.TITLE	DECISION
T	Brayshaw	+44214144778	Constraint-based reasoning	Y
J	Cunningham	+876352764763	Constraint-based reasoning	Y
L	Kwiatowska	+55324484703711	Constraint-based reasoning	Y
S	Beale	+2145995353214	Relational completeness of database sublanguages	Y
W	Edmondson	+44213313214	Relational completeness of database sublanguages	Y
M	Wallace	+447148942892	Relational completeness of database sublanguages	Y
S	Beale	+2145995353214	The Object-Oriented Database Manifesto	Y
T	Brayshaw	+44214144778	The Object-Oriented Database Manifesto	Y
J	Cunningham	+876352764763	The Object-Oriented Database Manifesto	Y
S	Beale	+2145995353214	The Relational Model of Data	Y
B	Dandy	+42484672394	The Relational Model of Data	Y
L	Kwiatowska	+55324484703711	The Relational Model of Data	Y

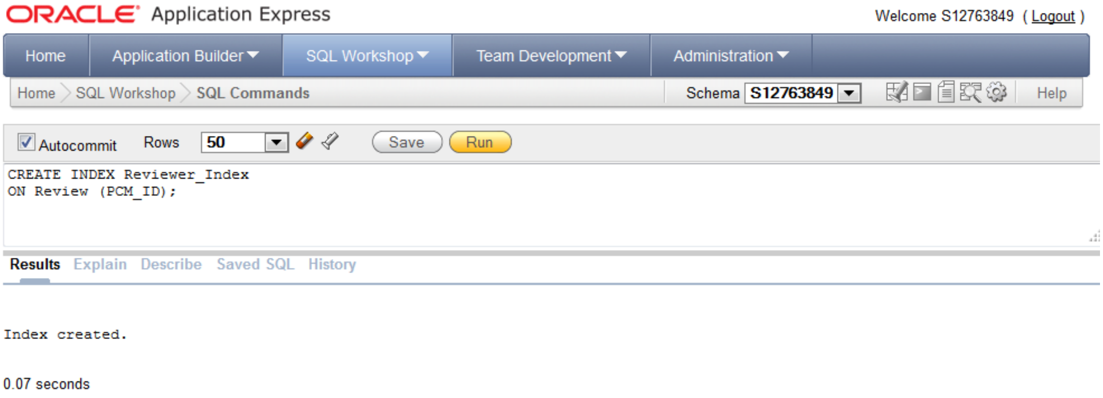
12 rows returned in 0.00 seconds   Download

Task 3

*Using SQL, create a suite of **indexes** on your new tables, turn the timer on and re-run the previous queries to generate a new set of times. Compare both indexed and non-indexed times.*

Atomic Index

```
CREATE INDEX Reviewer_Index
ON Review (PCM_ID);
```

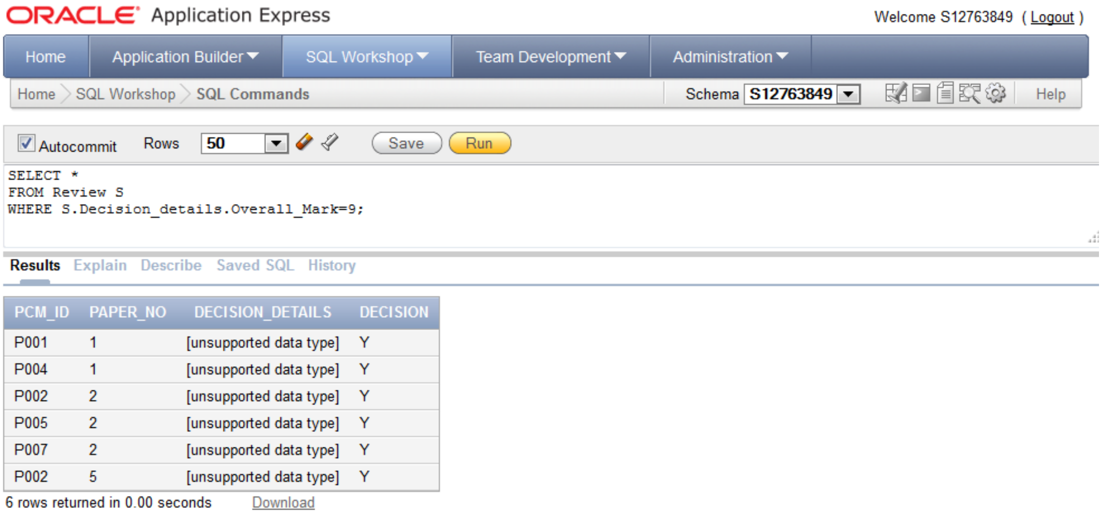


```
CREATE INDEX Reviewer_Index
ON Review (PCM_ID);
```

```
Index created.
```

0.07 seconds

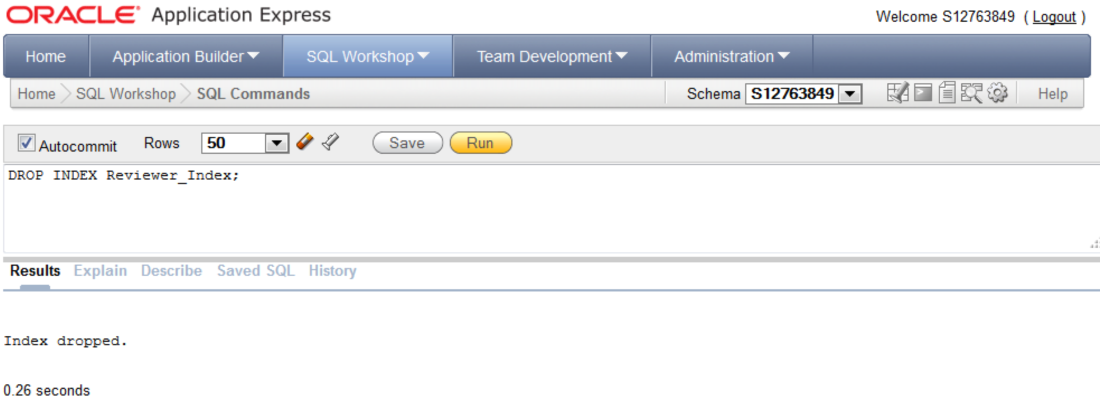
```
SELECT *
FROM Review S
WHERE S.Decision_details.Overall_Mark=9;
```



```
SELECT *
FROM Review S
WHERE S.Decision_details.Overall_Mark=9;
```

PCM_ID	PAPER_NO	DECISION_DETAILS	DECISION
P001	1	[unsupported data type]	Y
P004	1	[unsupported data type]	Y
P002	2	[unsupported data type]	Y
P005	2	[unsupported data type]	Y
P007	2	[unsupported data type]	Y
P002	5	[unsupported data type]	Y

6 rows returned in 0.00 seconds



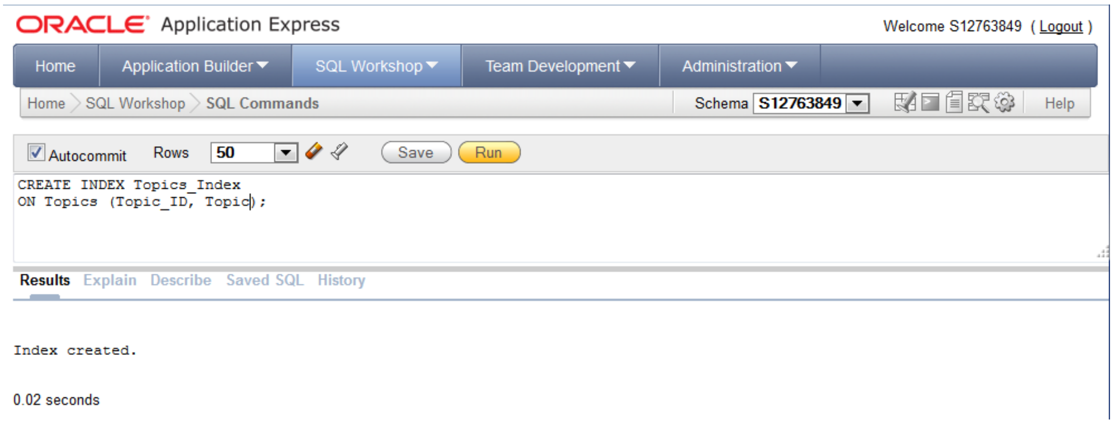
```
DROP INDEX Reviewer_Index;
```

```
Index dropped.
```

0.26 seconds

Composite Index

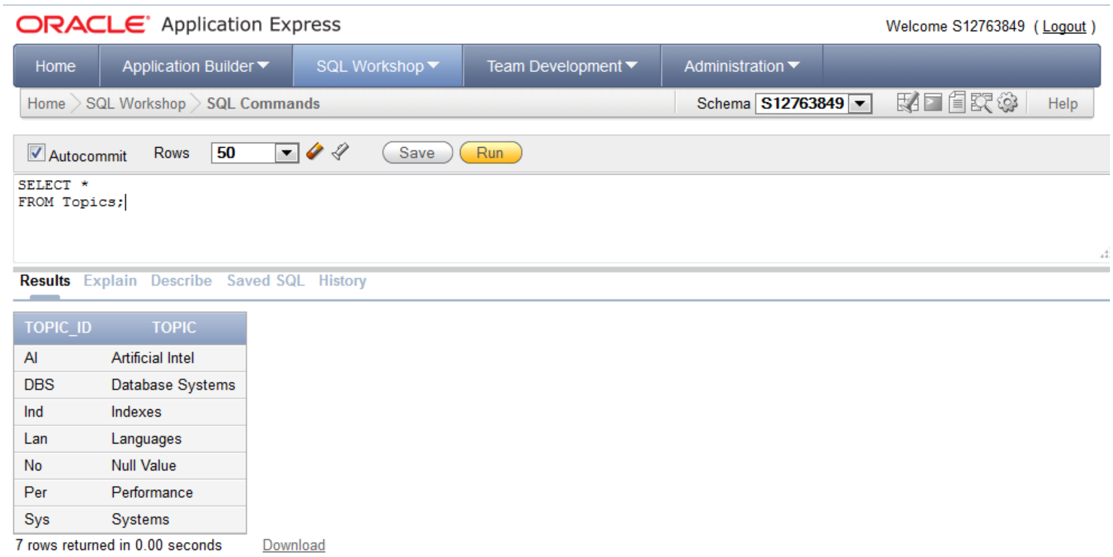
```
CREATE INDEX Topics_Index
ON Topics (Topic_ID, Topic);
```



```
CREATE INDEX Topics_Index
ON Topics (Topic_ID, Topic);
```

```
Index created.

0.02 seconds
```

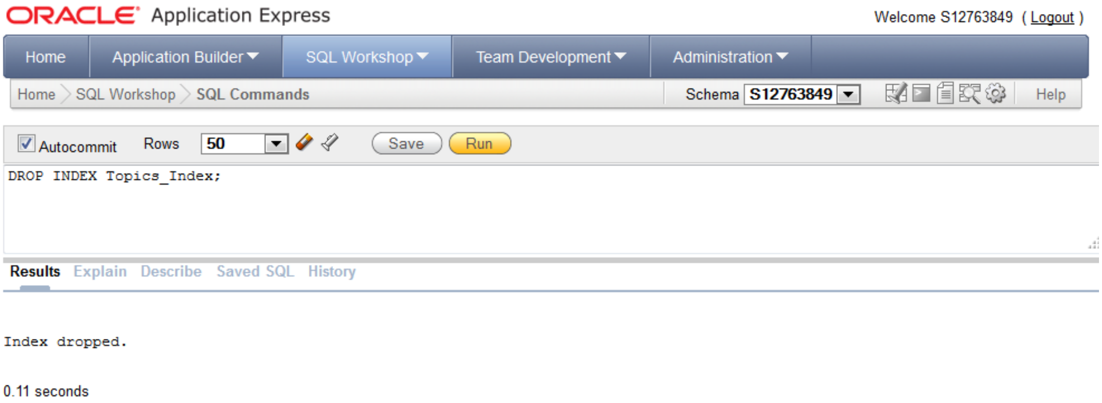
```
SELECT *
FROM Topics;
```



```
SELECT *
FROM Topics;
```

TOPIC_ID	TOPIC
AI	Artificial Intel
DBS	Database Systems
Ind	Indexes
Lan	Languages
No	Null Value
Per	Performance
Sys	Systems

7 rows returned in 0.00 seconds

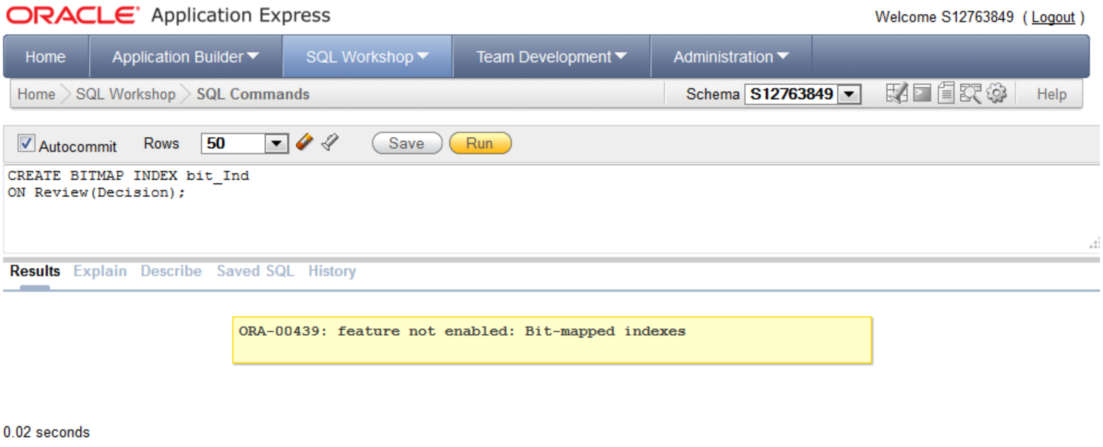


```
DROP INDEX Topics_Index;
```

```
Index dropped.

0.11 seconds
```

Bitmap Index

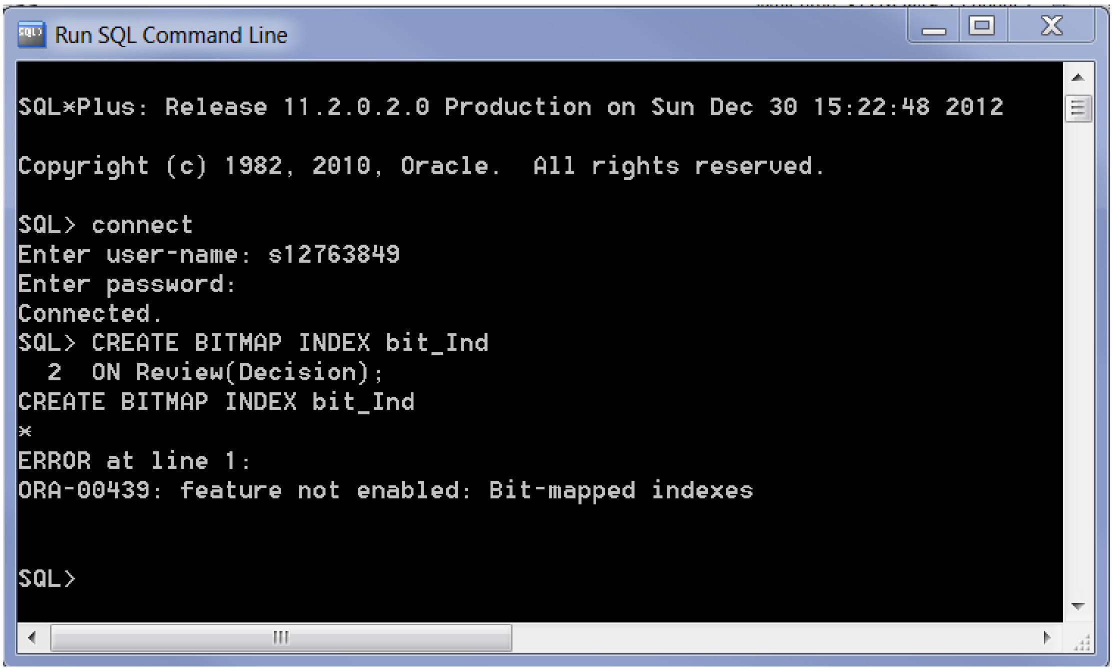
```
CREATE BITMAP INDEX bit_Ind
ON Review(Decision);
```



```
CREATE BITMAP INDEX bit_Ind
ON Review(Decision);
```

```
ORA-00439: feature not enabled: Bit-mapped indexes
```

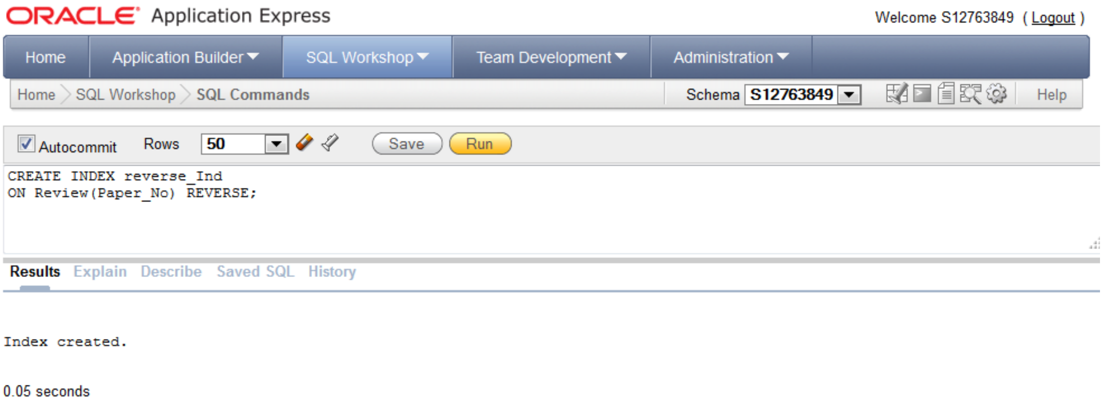
0.02 seconds



```
SQL*Plus: Release 11.2.0.2.0 Production on Sun Dec 30 15:22:48 2012

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

SQL> connect
Enter user-name: s12763849
Enter password:
Connected.
SQL> CREATE BITMAP INDEX bit_Ind
  2  ON Review(Decision);
CREATE BITMAP INDEX bit_Ind
*
ERROR at line 1:
ORA-00439: feature not enabled: Bit-mapped indexes


SQL>
```

Reverse Key Index

```
CREATE INDEX reverse_Ind
ON Review(Paper_No) REVERSE;
```

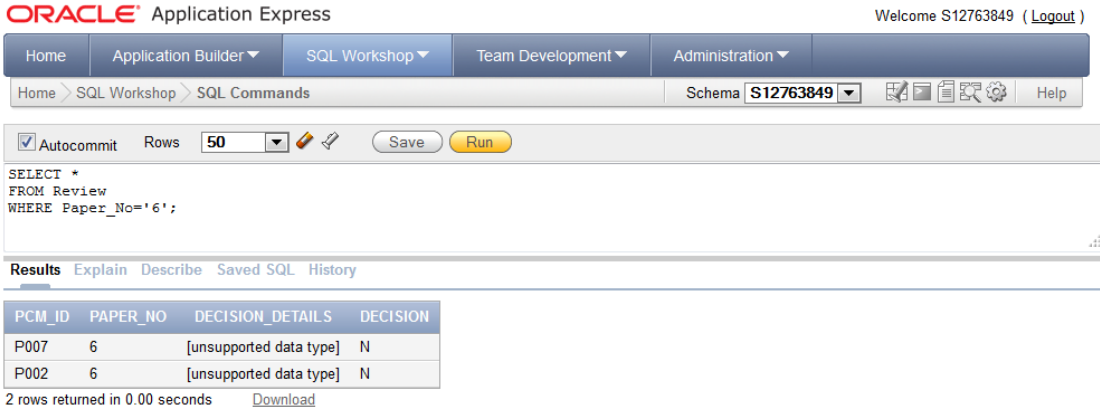


```
CREATE INDEX reverse_Ind
ON Review(Paper_No) REVERSE;
```

```
Index created.
```

0.05 seconds

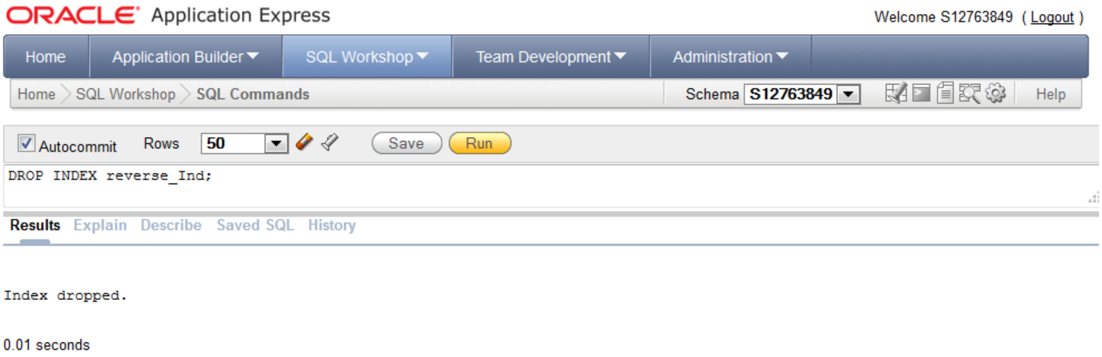
```
SELECT *
FROM Review
WHERE Paper_No='6';
```



```
SELECT *
FROM Review
WHERE Paper_No='6';
```

PCM_ID	PAPER_NO	DECISION_DETAILS	DECISION
P007	6	[unsupported data type]	N
P002	6	[unsupported data type]	N

2 rows returned in 0.00 seconds



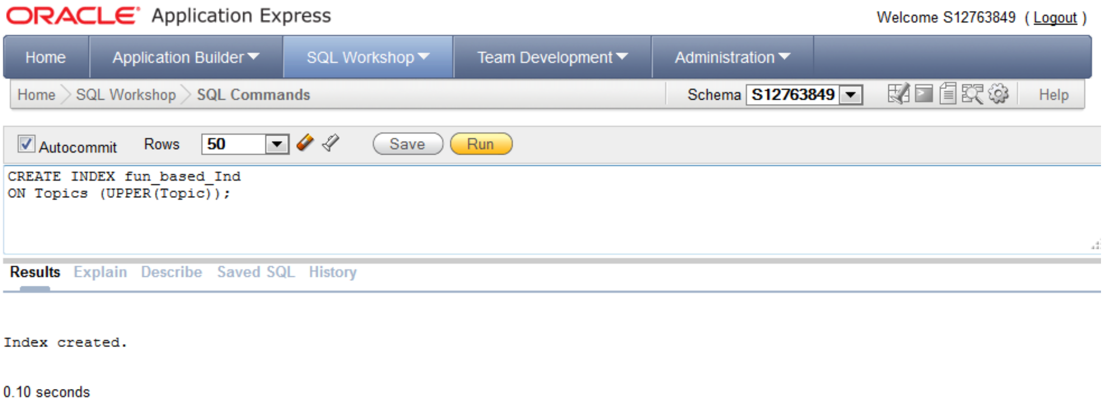
```
DROP INDEX reverse_Ind;
```

```
Index dropped.
```

0.01 seconds

Function-Based Index

```
CREATE INDEX fun_based_Ind
ON Topics (UPPER(Topic));
```

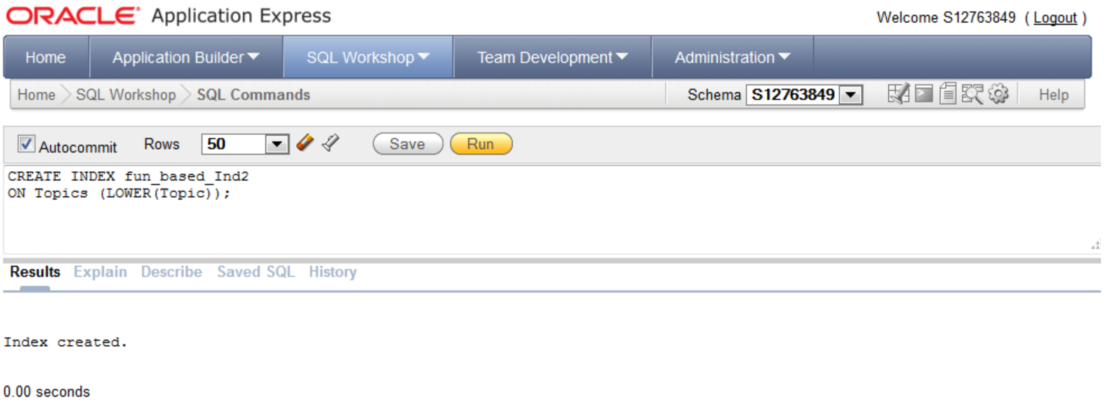


```
CREATE INDEX fun_based_Ind
ON Topics (UPPER(Topic));
```

```
Index created.
```

0.10 seconds

```
CREATE INDEX fun_based_Ind2
ON Topics (LOWER(Topic));
```

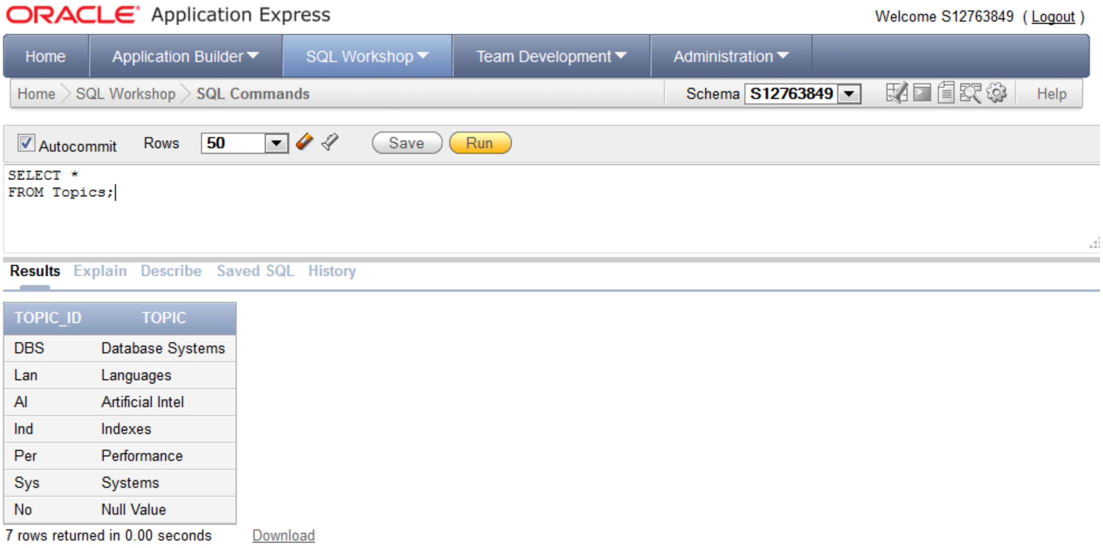


```
CREATE INDEX fun_based_Ind2
ON Topics (LOWER(Topic));
```

```
Index created.
```

0.00 seconds

```
SELECT *
FROM Topics;
```



```
SELECT *
FROM Topics;
```

TOPIC_ID	TOPIC
DBS	Database Systems
Lan	Languages
AI	Artificial Intel
Ind	Indexes
Per	Performance
Sys	Systems
No	Null Value

7 rows returned in 0.00 seconds

Task 4

*Explain how **clustering** and **partitioning** may also be utilized in the pursuit of query optimization. How do they differ from the indexing approach? Try and implement these concepts for your own tables.*

Indexing

The purpose of indexing is simply to provide a faster access to the results expected in a database query. Whilst the indexed results shown earlier in this paper do not particularly highlight this well, due to the relatively small amount and size of the tables used accompanied by the failure of APEX to currently work as expected with Object Orientated features, it is still true that indexed searches will in general produce results to SQL queries in a more timely fashion.

Should the tables used earlier have for instance thousands of papers, authors and reviews then it would be very accurate to suggest that the use of indexes would hugely increase the speed of searches recorded on them.

There are however also other techniques that can be used in the optimisation process, such as clustering and partitioning.

Clustering

"A cluster is a group of tables that share the same data blocks because they share common columns and are often used together." (Oracle, 2012)

Clusters are a method of storing data from multiple tables in the same place. A cluster is a natural combination of data that fits together due to its being in regular use as a collective. In the real world there are things that naturally fit together, such as *sea and sand*, and therefore are most often thought of as a single entity.

In the tables created there are many combinations of items that are often wanted to appear together at all times, such as;

- Paper Table and Author Table
- Review Table and Reviewer (PCM) Table

These items of data within the database can therefore be clustered together for future reference.

The purpose of clustering, like Indexing, is to make querying and therefore selection of data quicker. This is especially valuable in large databases with high levels of structure and hierarchy.

The main benefit of clustering over indexing of tables is that they use a **cluster key** system, where each cluster is given a cluster key value, which is stored only once. This means no matter how many different rows there are within a table less storage is required within the database to store the optimization value. Ergo a cluster requires and consumes less processing power and memory than a standard index. (Oracle, 2012).

Partitioning

Partitioning, whilst only suitable for *'Very Large Databases'*, is a process used in large tables where they are decomposed into smaller more manageable pieces. Each of these *'bite sized'* pieces is known as a partition. The purpose of such partitions is to that queries can be run on individual segments within a large table, once again to achieve optimization and make query results return more promptly.

Partitioning of large tables could be done in any number of manners, however if we take the tables already used in this paper and assume they were containing sufficient data we could suggest something such as partitioning the Reviews table by date, showing reviews for all different months of the year. Alternatively the Author (and/or PCM) table could be partitioned by geographic location.

The main benefit of this method over Indexing is that there become a huge amount of partition options possible. Therefore there are equally vast optimisation processes available to the Database Manager/Administrator, giving much greater flexibility in the creation and management of said database. It does this without the need to change the initial table at all in its creation, nor indeed change any of the SQL queries in their syntax. This makes partitioning and extremely powerful tool in the management of very large databases.

Task 5

Research and summarize in your own words the following current and future topics in database systems:

- *The impact of **XML** on relational databases*
- *The key techniques employed in database **security***
- *The use of **multi-media objects** in databases (GIS, digital maps etc)*

The impact of XML on relational databases

XML (eXtensible Markup Language) is described as “A meta-language (a language for describing other languages) that enables designers to create their own customized tags to provide functionality not available with HTML.” (Connolly & Begg, 2005).

A common misconception about XML however as stated by Steegmans et al (2004) is that XML can be used directly as a database. This is NOT the case; it is merely a language that can be used in conjunction with a database. This is especially valid when one considers that XML is indeed a very inefficient storage method in itself.

Since its ratification by W3C in 1998 XML has drastically changed computing. Primarily created as a programming language to give more flexibility than existing HTML, XML has uses ranging from graphical interfaces (GUIs), embedded systems, distributed systems and also now databases and database management. And due to its flexibility XML is now becoming the main medium used in businesses for the exchange of data.

The main reasons for this boom in the use of XML are down to its characteristics of being:

- ♦ Simple
- ♦ Open Standard
- ♦ Extensible
- ♦ Reusable
- ♦ Separates content and presentation
- ♦ Has improved load balancing
- ♦ Supports data integration
- ♦ Can describe data from a variety of applications
- ♦ More advanced search engines
- ♦ New Opportunities

We have seen earlier how relational databases store data in tables with traditional rows and columns. However this contrasts with XML and as Hunter (2007) explains, in XML documents order is intrinsically present, imposed by the criteria that define a well-formed XML document. This in turn can lead to the ability to store more data that in a relational database would be in a highly complex structure, in a more simple way using XML.

XML databases have a number of uses, however as Steegmans et al (2004) writes, the most common of these is to publish data stored in an existing database as XML on a web client.

As web clients intrinsically use the HTML protocol, as laid down in the OSI 7 Layer Model, or variations there-of, XML, being one of these variations, allows simple, efficient and flexibly replication and integration of database data onto a web client or application. XML can be used in two ways with databases in this scenario, either taking data from a database and storing it in an XML document for publishing on a web client or taking data from a web client to store in a database.

The process of taking data from a database and creating an XML document with it is known as publishing, whilst the process of taking data from an XML document and storing it in a database is known as shredding.

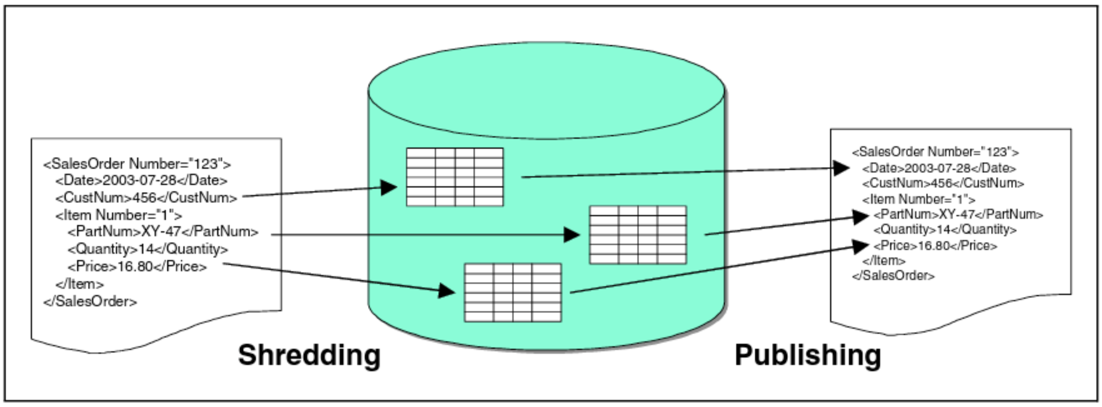


Figure 1: Taken from Steegmans et al (2004)

As Steegmans et al (2004) goes on to say another use of XML, that is becoming increasingly popular is to model semi-structured data. Due to XML's extensible nature as mentioned earlier, when databases are likely to change, potentially dramatically and/or regularly, XML becomes an excellent tool to accommodate such changing landscapes. Where traditional relational or even object orientated databases can be limiting due to their very nature as hierarchical complex structures the relative simplicity of XML can make for a better language to suit ever changing data.

Whilst XML does not and most likely will not replace databases or the need for DBMS, it is still an important computer language and as can be seen has much to offer the database world to add to and enhance those existing technologies already at use in relational databases.

The key techniques employed in database security

"A database is not an island. Most often it is a server deployed as a network node that provides persistence and transactional services to applications." (Natan, 2005)

Databases like any other device employed on a network require security, even more so when a database is being used to contain sensitive material. This material could be sensitive for data protection reasons, it could be sensitive for official secrets reasons or it could be sensitive for competitive reasons. There are whole manners of reasons why any data can be sensitive and wanted to be kept secure, and with databases being more and more a place where potentially sensitive materials (data) are kept, it is imperative for any business to keep their databases secure.

There is of course, in addition to this also a large proportion of data stored in databases that is wanted to be seen by the public, very often in a read only variance. However more and more, as databases become integrated with web clients, using language such as XML as previously discussed there is a plethora of information stored in databases where the public not only need to see it, but also actions made upon it need to be able to update the database. Take for instance the online shopping store, such as ebay or Amazon, where customers' orders need to update a database to show current stock levels. To this end customers' accounts need some sort of permission to perform this action. At the same time however, the store does not want customers to have permission to change the prices of items to suit themselves.

As Thuraisingham et al (2002) write, security provisions for computing focus on controlling access. The foundation for any security system is authorization and authentication. So for instance a user needs to firstly validate and ensure they are who they say they are and then that they are allowed to perform the operations that they wish to and are trying to perform.

SQL Databases such as Oracle, handle this type of security with the `GRANT` statement. In a basic database a System Administrator may create a range of users each with differing permissions, these permissions can be allocated on a one-by-one basis, with each individual being granted certain permissions. Alternatively users can be grouped themselves by role and permissions granted to specific roles occupied by individual users.

Typical permissions that can be granted for users or roles can be seen in the following table; overleaf. These privileges, like most things, can be granted and also revoked.

CONNECT	Allows a user to connect to the database
SELECT	Gives a user the right to access the table concerned
INSERT	Gives a user the right to add rows to a table
DELETE	Gives a user the right to remove rows from a table
UPDATE	Gives a user the right to change values in the table
INDEX	Gives a user the right to create and drop indexes on a table
ALTER	Gives the user the right to add new columns to a table

Figure 2: Table Privileges (Adapted from Van Der Lars, 1997)

We mentioned earlier however that there may be a vast amount of users to a database, take for instance a large company such as a national or global supermarket chain. They may have users such as IT managers and administrators that require far greater database access than warehouse staff or shop floor staff. There could be managers that require access to a completely different set of data for managerial or decision-making activities than those required by other staff. The same could be said of staff in an educational establishment, such as a college or university. Differing staff (and students) will often require differing access. The question then becomes, “*Does it really matter if all staff and customers can do everything to the database anyway?* “ The answer to this question is unequivocally *“Yes, it does.”* If all staff and/or customers have the same amount of access to a company’s database, then there is a very real danger to the integrity of the data stored within it. Integrity of data can easily be changed both by accident or deliberate manipulation. This therefore becomes another reason why it is imperative for a Database Manager to mange the security of a database.

As Ritchie (1998) states, the term integrity in the context of databases refers to the correctness and consistency of the data stored in the database. To this end not only does the data stored need to remain consistent with the relational set up of the database but it also needs to be correct. For example, if students in a University had access to their grades and permissions to alter them, there is a high chance this would inevitably happen. Databases therefore use permissions to keep security and make sure such eventualities cannot happen.

Also as mentioned earlier with the influx of XML and OO databases, and more and more companies using the Internet for sales, transactions are being done online via online web fronted databases. This term *transaction* also has a database connotation. As described by Ritchie (1998) a transaction “refers to a group of changes and/or queries to the database, which for the purpose of database integrity, must be performed as a single unit.”

All of the changes made within a transaction may lead to changes being required within a database updating a customer’s account details, purchase histories and producing billing information, to updating stock inventories, tax requirements for the company and staff sales achievements. There are numerous changes that could be instigated by a ‘simple’ online transaction. Any of these changes could lead to a database driven computer system’s failure. To this end security becomes again paramount to the correct and successful running of the business. And why correctly created and formatted permissions are crucial.

So taking a large company with numerous employees as discussed earlier, in order to grant the correct permissions to the correct person accessing the database, roles can be created. A role can be viewed as generic user, such as manager, director, sales staff, admin, HR, technical, customer etc, and each of these roles can be granted a requisite amount of permissions to access and change only certain parts of the database without damaging its integrity. Specific users, the people who will access the database, can then be given a role and its associated set of permissions.

In addition to this with many databases now being accessible online, or via a company’s intranet, database managers have to consider the threat of *hacking* into systems to either change sensitive values or gain access to sensitive information. TO this end firewalls are more and more being used to attempt to thwart or at least minimise this threat.

The use of multi-media objects in databases

Whilst traditional relational databases have been widely used in all facets of industry it is only with the creation of object orientated and object relational databases that applications such as CAD (Computer Aided Design), GIS (Geographic Information Systems) and multimedia storage systems can also interact with databases (Hernandez, 2003).

In traditional database management systems such as relational database systems, only textual and numerical data is stored and managed in the database and there is no need to consider the synchronicity among media. Retrieving data is of- ten based on simple comparisons of text or numerical values, which is no longer adequate for the multimedia data. (Chen, 2002) To this end object oriented data models have been suggested as a data model that does provide facilities to manage and store multimedia data.

There are now a new generation of databases using multimedia objects many of which are web and/or mobile application hosted, such as;

- The Internet Movie Database (http://www.imdb.com)
- Gracenote Database (http://www.gracenote.com) as used by iTunes
- Google Maps (https://maps.google.co.uk)
- Tineye Image Search (http://www.tineye.com)
- Getty Images (http://www.gettyimages.co.uk)

Take for instance the Apple iTunes store, which uses the Gracenote database to store literally tens of millions of music and video files (with associated art-works) centrally which can be accessed in most countries throughout the world.

> *"Gracenote touches hundreds of millions of music fans and TV viewers everyday. The Gracenote database is the largest source of music and video metadata on the planet, featuring descriptions of more than 130 million tracks and TV listings for 28 countries. The database receives more than 500 million queries everyday and more than 15 billion every month. If you measured Gracenote against a search engine, we would rank among the world's biggest." (Gracenote.com)*

By their very nature these multimedia data sources are often completely different from each other, therefore traditional databases may be able to store them but not produce any relevant query results on them, making them redundant for something such as Gracenote. A database such as this will need many more capabilities than a traditional relational database, as multimedia objects have these heterogeneous characteristics. A multimedia database system allows users to make queries for information very specific and related to both text, images or data. A further example of this could be Tineye’s (and Google’s) reverse image searching functions where an image can be uploaded and the database will be searched for same and even similar images. *“To date, TinEye has indexed 2,253,470,454 images from the web to help you find what you're looking for.”* (Tineye, 2013)

To access this new multimedia functionality of these databases traditional languages such as SQL are not particularly suitable, so new audiovisual and interactive query languages are being developed and honed. (Chen, 2002)

Again in the case of Tineye, they use XML and imagemap files to create the additional information that is stored with the files to allow query or search capabilities.

Tineye XML imagemap files have the following details, as a minimum:

- A unique ID for the image
- A single URL for the page that displays the image (a link to this page will be displayed by TinEye)
- An image URL, for us to download an unwatermarked version of the image

(Tineye, 2013)

An imagemap file for "example.com" might look like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<tineye-list
   creation-date="2010-04-20T11:48:43.0Z"
   xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"

xs:noNamespaceSchemaLocation="http://www.tineye.com/contributing/imag
emap.xsd">
   <image>
      <id>SN0982345</id>
      <page-url>http://www.example.com/image/SN0982345</page-url>
      <image-url>https://comps.example.com/comp/SN0982345.jpg</image-
url>
      <author-info>
         <author-id>jdoe</author-id>
         <author-name>Jane Doe</author-name>
      </author-info>
      <keywords lang='en'>cat, sun, sleep</keywords>
   </image>
   <image>
      <id>SN08972345</id>
      <page-url>http://www.example.com/image/SN08972345</page-url>
      <image-
url>https://comps.example.com/comp/SN08972345.jpg</image-url>
      <author-info>
         <author-id>jdoe2</author-id>
         <author-name>Jack Doe</author-name>
      <author-info>
      <metadata>
         <width>12000</width>
         <height>8000</height>
         <file-size>12373020</file-size>
      </metadata>
      <keywords lang='en-US'>dog, fire plug, sidewalk</keywords>
      <keywords lang='en-GB'>dog, fire hydrant, pavement</keywords>
      <keywords lang='de'>hund, hydrant, b&#252;rgersteig</keywords>
   </image>
</tineye-list>
```

(Tineye, 2013)

In these new audiovisual and interactive query languages queries are represented as *words* and are operated on both the attributes of a multimedia file along with metadata and their content.

References

Chen, S-C et al. 2002. Semantic Models for Multimedia Database Searching and Browsing. Boston: Kluwer Academic Publishing.

Connolly, T & Begg, C. 2005. Database Systems: A Practical Approach to Design, Implementation and Management. 4th Ed. Harlow: Pearson Education Ltd.

Gracenote. 2012. Gracenote By the Numbers [online]. Available at: http://www.gracenote.com/company [Accessed on 11th Jan 2013]

Hernandez, M. 2003. Database Design for Mere Mortals. 2nd Ed. Boston: Pearson Education.

Hunter, D et al. 2007. Beginning XML. 4th Ed. Indianapolis: Wiley Publishing,

Natan, B. 2005. Implementing Database Security and Auditing. Oxford: Elsevier Digital Press.

Oracle. 2012. Oracle® Database Concepts, 11g Release 1 (11.1) [Online]. Available at: http://www.oracle.com/pls/db111/homepage [accessed on 28th Dec 2012].

Ritchie, C. 1998. Relational Database Principles. London: Letts Educational.

Steegmans, B et al. 2004. XML for DB2 Information Integration. IBM: Redbooks.

Thuraisingham et al. 2002. Data and Applications Security: Developments and Directions. 2nd Ed. Boston: Kulwer Academic Publishers.

Tineye. 2013. Tineye: About [online]. Available at: http://www.tineye.com/about [Accessed on 12th Jan 2013]

Tineye. 2013. Tineye: Image Submission [online]. Available at: http://www.tineye.com/imagesubmission [Accessed on 12th Jan 2013]