Tables:

One of the most important tables in Oracle Fusion (Receivables) is RA\_CUSTOMER\_TRX\_ALL. This table stores the header information for all receivables

transactions, including invoices, debit memos, and credit memos. Key columns in this table include:

• CUSTOMER\_TRX\_ID: The primary key for the transaction.

• TRX\_NUMBER: The transaction number.

• BILL\_TO\_CUSTOMER\_ID: The ID of the customer being billed.

• TRX\_DATE: The date of the transaction.

The RA\_CUSTOMER\_TRX\_ALL table in Oracle Fusion is crucial for managing invoice, debit memo, credit memo, and bills receivable information.

Here are some of the most important columns that are frequently used:

1. CUSTOMER\_TRX\_ID: The unique identifier for each transaction.

2. TRX\_NUMBER: The transaction number.

3. TRX\_DATE: The date of the transaction.

4. BILL\_TO\_CUSTOMER\_ID: The identifier for the billing customer.

5. SOLD\_TO\_CUSTOMER\_ID: The identifier for the customer to whom the goods or services are sold.

6. SHIP\_TO\_CUSTOMER\_ID: The identifier for the customer to whom the goods are shipped.

7. SET\_OF\_BOOKS\_ID: Identifies the ledger associated with the transaction.

8. CREATION\_DATE: The date and time when the row was created.

9. LAST\_UPDATE\_DATE: The date and time when the row was last updated.

10. LAST\_UPDATED\_BY: The user who last updated the row

=============================================================================

IMPORTANT TABLE IN ORACLE DOCS UNDER RECIEVABLE SECTION

----------------------------------------------------------

day-to-day accounts operations. Billing, Receivables Balances, Revenue Management, and Credit Management.

• AR\_CMGT\_CREDIT\_DATA\_INT

This table contains the imported credit data.

Pk: CREDIT\_DATA\_ID This column uniquely identifies each credit data entry, making it essential for tracking

PARTY\_ID: This column uniquely identifies the customer associated with the credit data, which is crucial for linking credit information to specific customers.

• AR\_PAYMENT\_SHEDULES\_ALL

Tracks payment schedules for transactions.

PAYMENT\_SCHEDULE\_ID: this is a primary key that tracks payment schedule entry,

Amount\_due\_remaining : Indicate remaining amount due , easy to understand outstanding amount

AR\_CASH\_RECIEPTS / RECOS /

CASH\_RECEIPT\_ID: this is a primary key which mainly tracks entry of cash receipts

Amount : amount of the cash receipts.

Tracks cash receipt details.

• RA\_CUSTOMER\_TRX\_LINES\_ALL

This table stores line information about invoices, debit memos, credit memos, and bills receivable.

PK:

1. CUSTOMER\_TRX\_LINE\_ID: The primary key for the table, uniquely identifying each transaction line.

2. CUSTOMER\_TRX\_ID: A foreign key linking to the RA\_CUSTOMER\_TRX\_ALL table, which stores the header information for the transaction.

3. LINE\_TYPE: Indicates the type of line, such as 'LINE' for regular invoice lines, 'TAX' for tax lines, 'FREIGHT' for freight lines, and 'CHARGES' for other charges.

4. DESCRIPTION: A description of the transaction line.

5. QUANTITY\_ORDERED: The quantity of the product ordered.

6. QUANTITY\_INVOICED: The quantity of the product invoiced.

7. UNIT\_SELLING\_PRICE: The selling price per unit for the transaction line.

8. EXTENDED\_AMOUNT: The total amount for the transaction line, calculated as quantity multiplied by the unit selling price.

A diagram of a company

AI-generated content may be incorrect.

A red line with white text

AI-generated content may be incorrect.

1. **HZ\_CUST\_ACCOUNTS**: This table stores information about customer accounts. Each customer account is associated with a party and can have multiple sites and contacts.

Party site

1. **HZ\_CUST\_ACCT\_SITES\_ALL**: Contains information about customer account sites, including site usage (e.g., bill-to, ship-to).

Party site uses

1. **HZ\_CUST\_SITE\_USES\_ALL**: Stores the specific uses of customer account sites, such as billing or shipping purposes.

**BANK:**

**AP\_BANKS**

**\_BANK\_BRANCHES**

**AP\_BANK\_ACCOUNT**

TCA TABLES IN HCM:

A close-up of a piece of paper

AI-generated content may be incorrect.

A close up of a piece of paper

AI-generated content may be incorrect.

A paper with writing on it

AI-generated content may be incorrect.

A white text with black text

AI-generated content may be incorrect.

A white text with black text

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

A white text with black text

AI-generated content may be incorrect.

**Basic Tables in Fusion HCM and which tables are affected when certain actions are performed:**

Oracle Fusion HCM has several important tables that store HR-related data. These tables are crucial for managing employees, payroll, benefits, and other HR processes. Here’s a detailed breakdown:

**### \*\*1. Key Tables in Oracle Fusion HCM\*\***

**#### \*\*Person Data Tables\*\***

- \*\*PER\_ALL\_PEOPLE\_F\*\* – Stores employee details like name, address, contact info, and employment status.

- \*\*PER\_PERSON\_NAMES\_F\*\* – Contains historical and current names of employees.

- \*\*PER\_ADDRESSES\_F\*\* – Stores employee addresses.

- \*\*PER\_EMAIL\_ADDRESSES\*\* – Maintains email details for employees.

- \*\*PER\_PHONES\*\* – Stores phone numbers and contact details.

**#### \*\*Assignment Tables\*\***

- \*\*PER\_ALL\_ASSIGNMENTS\_F\*\* – Tracks job, position, department, location, and compensation details.

- \*\*PER\_ASSIGNMENT\_SUPERVISORS\_F\*\* – Stores supervisor details for each assignment.

- \*\*PER\_ASSIGN\_WORK\_MEASURES\_F\*\* – Tracks work measurement details like hours worked.

**#### \*\*Payroll Tables\*\***

- \*\*PAY\_ELEMENT\_ENTRIES\_F\*\* – Stores salary, bonus, and allowances.

- \*\*PAY\_PAYROLL\_ACTIONS\_F\*\* – Tracks payroll actions like hire, transfer, promotion, and termination.

- \*\*PAY\_RUN\_RESULTS\_F\*\* – Stores payroll run results for employees.

**#### \*\*Benefits Tables\*\***

- \*\*BEN\_PERSON\_COVERAGES\*\* – Stores employee benefits like health insurance and retirement plans.

- \*\*BEN\_ENROLLMENTS\*\* – Tracks enrollment status of employees in benefit plans.

**#### \*\*Absence Management Tables\*\***

- \*\*PER\_ABSENCE\_ATTENDANCES\*\* – Records employee leave and attendance details.

- \*\*PER\_PERIODS\_OF\_SERVICE\*\* – Stores details about an employee’s service periods.

**#### \*\*Job and Position Tables\*\***

- \*\*PER\_JOBS\*\* – Stores job definitions, including job titles, grades, and pay ranges.

- \*\*PER\_POSITION\_DEFINITIONS\*\* – Contains position-related information like hierarchy and type.

**### \*\*2. How Tables Are Affected by Actions\*\***

**#### \*\*Hiring a New Employee\*\***

- \*\*PER\_ALL\_PEOPLE\_F\*\* – A new record is created with employee details.

- \*\*PER\_ALL\_ASSIGNMENTS\_F\*\* – A new assignment is added.

- \*\*PAY\_ELEMENT\_ENTRIES\_F\*\* – Salary details are recorded.

- \*\*BEN\_PERSON\_COVERAGES\*\* – Benefits enrollment is updated.

**#### \*\*Updating Employee Contact Information\*\***

- \*\*PER\_PHONES\*\* – Phone number changes are recorded.

- \*\*PER\_EMAIL\_ADDRESSES\*\* – Email updates are stored.

- \*\*PER\_ADDRESSES\_F\*\* – Address changes are tracked.

**#### \*\*Processing Payroll\*\***

- \*\*PAY\_PAYROLL\_ACTIONS\_F\*\* – Payroll process is initiated.

- \*\*PAY\_RUN\_RESULTS\_F\*\* – Payroll calculations are stored.

- \*\*PAY\_ELEMENT\_ENTRIES\_F\*\* – Salary components are updated.

**#### \*\*Employee Termination\*\***

- \*\*PER\_PERIODS\_OF\_SERVICE\*\* – Service period is closed.

- \*\*PER\_ALL\_ASSIGNMENTS\_F\*\* – Assignment is ended.

- \*\*PAY\_PAYROLL\_ACTIONS\_F\*\* – Final payroll processing occurs.

A screenshot of a computer program

AI-generated content may be incorrect.

Sure! Let's break down the Trading Community Architecture step-by-step in simpler terms:

**Basic Concepts**

1. **Party**: This is any entity (like a person or organization) involved in trading.
2. **Sites**: Each party can have multiple locations or sites.
3. **Addresses**: Each site has an address.
4. **Purpose**: Each site has a specific purpose, such as receiving money.

**Key Tables and Their Roles**

1. **HZ\_PARTIES**: Stores information about each party.
2. **HZ*PARTY*SITES**: Stores information about each site of the party.
3. **HZ\_LOCATIONS**: Contains the actual address details linked to the site.
4. **HZ*PARTY*SITE\_USES**: Stores the purpose of each site (e.g., billing, shipping).
5. **HZ*CONTACT*POINTS**: Stores communication details for sites or persons.
6. **HZ*PARTY*RELATIONSHIPS**: Stores relationships between parties (e.g., parent-child relationships).
7. **HZ*CUST*ACCT**: Stores customer details. It links the party ID but does not store the customer account ID in HZ\_PARTIES.
8. **HZ*CUST*ACCT\_SITES**: Stores information about customer account sites.
9. **HZ*CUST*ACCT*SITES*USES**: Stores the purpose of customer account sites.
10. **POZ\_SUPPLIERS**: Stores supplier details.
11. **POZ*SUPPLIERS*SITES**: Stores flags related to the purchasing side of suppliers.
12. **AP\_BANKS**: Stores bank details.
13. **AP*BANK*BRANCHES**: Stores details about bank branches.
14. **AP*BANK*ACCT**: Stores bank account details, not directly related to AP\_BANK.
15. **FOR*ALL*PEOPLE\_F**: Stores information about people who can have addresses.
16. **FOR*ADDRESSES*F**: Stores address details.
17. **FOR*ADDRESS*USAGES\_F**: Stores the usage details of addresses.

**How It Works Together**

* **Parties** have multiple **sites**.
* Each **site** has an **address** stored in **HZ\_LOCATIONS**.
* The **purpose** of each site is stored in **HZ*PARTY*SITE\_USES**.
* **Contacts** for each site are stored in **HZ*CONTACT*POINTS**.
* **Relationships** between parties are stored in **HZ*PARTY*RELATIONSHIPS**.
* **Customer details** are stored in **HZ*CUST*ACCT**, linking to the party ID.
* **Supplier details** and their purchasing flags are stored in **POZ*SUPPLIERS*** *and* ***POZ*SUPPLIERS\_SITES**.
* **Bank details** are stored in **AP*BANKS*** *and* ***AP*BANK\_BRANCHES**.
* **Bank account details** are stored in **AP*BANK*ACCT**.
* **People** and their addresses are stored in **FOR*ALL*PEOPLE*F*** *and* ***FOR*ADDRESSES\_F**.
* **Address usage** details are stored in **FOR*ADDRESS*USAGES\_F**.

**Summary**

This architecture helps manage and organize information about parties, their sites, addresses, purposes, contacts, relationships, customer accounts, suppliers, banks, and people in a structured way.

**Error Tables:**

A screenshot of a computer error

AI-generated content may be incorrect.

**Table Overview**

The table lists different types of data sheets and their corresponding interface and error tables. Here's what each column means:

* **Sheet Name**: The name of the data sheet you are working with.
* **Interface Table**: The table where data from the sheet is uploaded.
* **Error Table**: The table where errors related to the data upload are recorded.

**Table Details**

| **Sheet Name** | **Interface Table** | **Error Table** |
| --- | --- | --- |
| Customers | AR\_*UPD\_*CUSTOMERS | AR\_*UPD\_*CUST\_ERRORS |
| Contacts | AR\_*UPD\_*CONTACTS | AR\_*UPD\_*CONT\_ERRORS |
| Reference Accounts | AR\_*UPD\_*REF\_ACCOUNTS | AR\_*UPD\_*REFACT\_ERRORS |
| Customer Bank Accounts | AR\_*UPD\_*BANK\_ACCOUNTS | AR\_*UPD\_*BNKACCT\_ERRORS |

**Process Monitoring**

When you upload customer data, the process can have one of four statuses:

1. **Succeeded**: Everything is complete and successful.
2. **Completed**: Not all child processes are complete; check for errors.
3. **Errors**: There are errors in one or more child processes.
4. **No records transferred**: Issues in processing data from one or more spreadsheets.

**Error Correction**

If there are errors, they might be related to specific fields like:

* Profile class
* Payment method
* Contact point email address
* Account site use code
* Bank account number

These errors indicate invalid data in these fields.

**Summary**

* **Monitor the upload process** to ensure all data is correctly transferred.
* **Check the error table** if the process status is not "Succeeded."
* **Correct any errors** in the specified fields and resubmit the data.

Sure! Here are some SQL queries for each table you mentioned:

### \*\*1. RA\_CUSTOMER\_TRX\_ALL\*\* – Retrieve transaction details

```sql

SELECT CUSTOMER\_TRX\_ID, TRX\_NUMBER, TRX\_DATE, BILL\_TO\_CUSTOMER\_ID

FROM RA\_CUSTOMER\_TRX\_ALL

WHERE TRX\_DATE BETWEEN '01-JAN-2025' AND '31-DEC-2025';

```

### \*\*2. AR\_CMGT\_CREDIT\_DATA\_INT\*\* – Retrieve imported credit data

```sql

SELECT CREDIT\_DATA\_ID, PARTY\_ID

FROM AR\_CMGT\_CREDIT\_DATA\_INT

WHERE PARTY\_ID = 12345;

```

### \*\*3. AR\_PAYMENT\_SCHEDULES\_ALL\*\* – Retrieve payment schedule details

```sql

SELECT PAYMENT\_SCHEDULE\_ID, AMOUNT\_DUE\_REMAINING

FROM AR\_PAYMENT\_SCHEDULES\_ALL

WHERE AMOUNT\_DUE\_REMAINING > 0;

```

### \*\*4. RA\_CUSTOMER\_TRX\_LINES\_ALL\*\* – Retrieve transaction line details

```sql

SELECT CUSTOMER\_TRX\_LINE\_ID, CUSTOMER\_TRX\_ID, LINE\_TYPE, DESCRIPTION, UNIT\_SELLING\_PRICE, EXTENDED\_AMOUNT

FROM RA\_CUSTOMER\_TRX\_LINES\_ALL

WHERE LINE\_TYPE = 'LINE';

```

### \*\*5. HZ\_CUST\_ACCOUNTS\*\* – Retrieve customer account details

```sql

SELECT CUSTOMER\_ID, ACCOUNT\_NUMBER

FROM HZ\_CUST\_ACCOUNTS

WHERE CUSTOMER\_ID = 98765;

```

### \*\*6. HZ\_CUST\_ACCT\_SITES\_ALL\*\* – Retrieve customer account site details

```sql

SELECT SITE\_ID, SITE\_USAGE

FROM HZ\_CUST\_ACCT\_SITES\_ALL

WHERE SITE\_USAGE = 'BILL\_TO';

```

### \*\*7. AP\_BANKS\*\* – Retrieve bank details

```sql

SELECT BANK\_NAME, BANK\_ID

FROM AP\_BANKS;

```

### \*\*8. PER\_ALL\_PEOPLE\_F\*\* – Retrieve employee details

```sql

SELECT PERSON\_ID, NAME, DATE\_OF\_BIRTH

FROM PER\_ALL\_PEOPLE\_F

WHERE DATE\_OF\_BIRTH > '01-JAN-1990';

```

### \*\*9. PAY\_ELEMENT\_ENTRIES\_F\*\* – Retrieve payroll details

```sql

SELECT ELEMENT\_ENTRY\_ID, SALARY, BONUS

FROM PAY\_ELEMENT\_ENTRIES\_F

WHERE SALARY > 50000;

```

### \*\*10. HZ\_PARTIES\*\* – Retrieve trading community party details

```sql

SELECT PARTY\_ID, PARTY\_NAME, PARTY\_TYPE

FROM HZ\_PARTIES;

```

Of course! Here are more SQL queries across different tables for various use cases:

### \*\*1. Fetch the latest transactions from RA\_CUSTOMER\_TRX\_ALL\*\*

```sql

SELECT CUSTOMER\_TRX\_ID, TRX\_NUMBER, TRX\_DATE, BILL\_TO\_CUSTOMER\_ID

FROM RA\_CUSTOMER\_TRX\_ALL

ORDER BY TRX\_DATE DESC

FETCH FIRST 10 ROWS ONLY;

```

### \*\*2. Find overdue payments from AR\_PAYMENT\_SCHEDULES\_ALL\*\*

```sql

SELECT PAYMENT\_SCHEDULE\_ID, AMOUNT\_DUE\_REMAINING, DUE\_DATE

FROM AR\_PAYMENT\_SCHEDULES\_ALL

WHERE DUE\_DATE < SYSDATE AND AMOUNT\_DUE\_REMAINING > 0;

```

### \*\*3. List customers who have made payments from AR\_CASH\_RECEIPTS\*\*

```sql

SELECT DISTINCT C.CUSTOMER\_ID, C.CUSTOMER\_NAME

FROM AR\_CASH\_RECEIPTS CR

JOIN HZ\_CUST\_ACCOUNTS C ON CR.CUSTOMER\_ID = C.CUSTOMER\_ID

WHERE CR.AMOUNT > 0;

```

### \*\*4. Retrieve invoice details for a specific customer from RA\_CUSTOMER\_TRX\_LINES\_ALL\*\*

```sql

SELECT CUSTOMER\_TRX\_ID, LINE\_TYPE, DESCRIPTION, EXTENDED\_AMOUNT

FROM RA\_CUSTOMER\_TRX\_LINES\_ALL

WHERE CUSTOMER\_TRX\_ID IN (

SELECT CUSTOMER\_TRX\_ID FROM RA\_CUSTOMER\_TRX\_ALL WHERE BILL\_TO\_CUSTOMER\_ID = 12345

);

```

### \*\*5. Find employees with pending payroll payments from PAY\_RUN\_RESULTS\_F\*\*

```sql

SELECT P.PERSON\_ID, P.NAME, PRF.PAYROLL\_PERIOD, PRF.AMOUNT

FROM PER\_ALL\_PEOPLE\_F P

JOIN PAY\_RUN\_RESULTS\_F PRF ON P.PERSON\_ID = PRF.PERSON\_ID

WHERE PRF.STATUS = 'PENDING';

```

### \*\*6. Identify employees assigned to a specific department from PER\_ALL\_ASSIGNMENTS\_F\*\*

```sql

SELECT PERSON\_ID, ASSIGNMENT\_ID, JOB\_ID, DEPARTMENT\_ID

FROM PER\_ALL\_ASSIGNMENTS\_F

WHERE DEPARTMENT\_ID = 'HR001';

```

### \*\*7. Find suppliers associated with specific bank accounts from POZ\_SUPPLIERS and AP\_BANK\_ACCOUNTS\*\*

```sql

SELECT S.SUPPLIER\_NAME, B.BANK\_ACCOUNT\_NUMBER

FROM POZ\_SUPPLIERS S

JOIN AP\_BANK\_ACCOUNTS B ON S.SUPPLIER\_ID = B.SUPPLIER\_ID;

```

### \*\*8. Retrieve customers who have errors in their accounts from AR\_UPD\_CUST\_ERRORS\*\*

```sql

SELECT CUSTOMER\_ID, ERROR\_MESSAGE

FROM AR\_UPD\_CUST\_ERRORS;

```